THE UNIVERSITY OF WARWICK

Library Declaration and Deposit Agreement

1. STUDENT DETAILS

Please complete the following:

Full name: Nikolas Landia

University ID number: 0320567

2. THESIS DEPOSIT

2.1 I understand that under my registration at the University, I am required to deposit my thesis with the University in BOTH hard copy and in digital format. The digital version should normally be saved as a single pdf file.

2.2 The hard copy will be housed in the University Library. The digital version will be deposited in the University's Institutional Repository (WRAP). Unless otherwise indicated (see 2.3 below) this will be made openly accessible on the Internet and will be supplied to the British Library to be made available online via its Electronic Theses Online Service (EThOS) service.
[At present, theses submitted for a Master's degree by Research (MA, MSc, LLM, MS or MMedSci) are not being deposited in WRAP and not being made available via EthOS. This may change in future.]

2.3 In exceptional circumstances, the Chair of the Board of Graduate Studies may grant permission for an embargo to be placed on public access to the hard copy thesis for a limited period. It is also possible to apply separately for an embargo on the digital version. (Further information is available in the Guide to Examinations for Higher Degrees by Research.)

2.4 *If you are depositing a thesis for a Master's degree by Research, please complete section (a) below.* For all other research degrees, please complete both sections (a) and (b) below:

(a) Hard Copy

I hereby deposit a hard copy of my thesis in the University Library to be made publicly available to readers ~~(please delete as appropriate) EITHER~~ immediately ~~OR after an embargo period of ..... months/years as agreed by the Chair of the Board of Graduate Studies.~~

I agree that my thesis may be photocopied. YES / ~~NO~~ (Please delete as appropriate)

(b) Digital Copy

I hereby deposit a digital copy of my thesis to be held in WRAP and made available via EThOS.

Please choose one of the following options:

~~EITHER~~ My thesis can be made publicly available online. YES / ~~NO~~ (Please delete as appropriate)

~~OR My thesis can be made publicly available only after.....[date] (Please give date)~~
~~YES / NO (Please delete as appropriate)~~

~~OR My full thesis cannot be made publicly available online but I am submitting a separately identified additional, abridged version that can be made available online.~~
~~YES / NO (Please delete as appropriate)~~

~~OR My thesis cannot be made publicly available online. YES / NO (Please delete as appropriate)~~

3. GRANTING OF NON-EXCLUSIVE RIGHTS

Whether I deposit my Work personally or through an assistant or other agent, I agree to the following:

Rights granted to the University of Warwick and the British Library and the user of the thesis through this agreement are non-exclusive. I retain all rights in the thesis in its present version or future versions. I agree that the institutional repository administrators and the British Library or their agents may, without changing content, digitise and migrate the thesis to any medium or format for the purpose of future preservation and accessibility.

4. DECLARATIONS

(a) I DECLARE THAT:

- I am the author and owner of the copyright in the thesis and/or I have the authority of the authors and owners of the copyright in the thesis to make this agreement. Reproduction of any part of this thesis for teaching or in academic or other forms of publication is subject to the normal limitations on the use of copyrighted materials and to the proper and full acknowledgement of its source.
- The digital version of the thesis I am supplying is the same version as the final, hard-bound copy submitted in completion of my degree, once any minor corrections have been completed.
- I have exercised reasonable care to ensure that the thesis is original, and does not to the best of my knowledge break any UK law or other Intellectual Property Right, or contain any confidential material.
- I understand that, through the medium of the Internet, files will be available to automated agents, and may be searched and copied by, for example, text mining and plagiarism detection software.

(b) IF I HAVE AGREED (in Section 2 above) TO MAKE MY THESIS PUBLICLY AVAILABLE DIGITALLY, I ALSO DECLARE THAT:

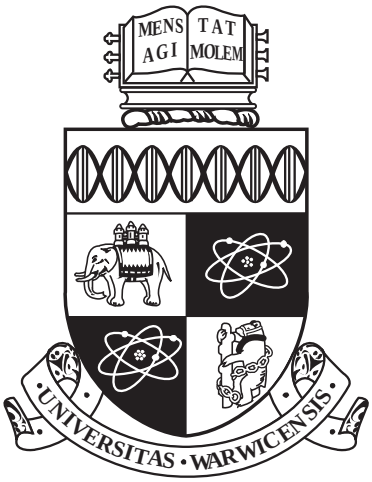
- I grant the University of Warwick and the British Library a licence to make available on the Internet the thesis in digitised format through the Institutional Repository and through the British Library via the EThOS service.
- If my thesis does include any substantial subsidiary material owned by third-party copyright holders, I have sought and obtained permission to include it in any version of my thesis available in digital format and that this permission encompasses the rights that I have granted to the University of Warwick and to the British Library.

5. LEGAL INFRINGEMENTS

I understand that neither the University of Warwick nor the British Library have any obligation to take legal action on behalf of myself, or other rights holders, in the event of infringement of intellectual property rights, breach of contract or of any other right, in the thesis.

Please sign this agreement and return it to the Graduate School Office when you submit your thesis.

Student's signature: .  ..... Date: 12/08/2013 .....



Content-Awareness and Graph-Based Ranking for Tag Recommendation in Folksonomies

by

Nikolas Landia

Thesis

Submitted to the University of Warwick

for the degree of

Doctor of Philosophy

Department of Computer Science

April 2013

THE UNIVERSITY OF
WARWICK

Contents

List of Tables

List of Figures

Acknowledgments

First and foremost I thank my supervisors Sarabjot Singh Anand and Nathan Griffiths for their invaluable guidance in completing this work. I have very much enjoyed our long and animated discussions and am thankful for their great knowledge, expertise and patience over the course of my studies. I would also like to thank my family, my parents and grandparents, whose loving support has helped a great deal during the more stressful times of writing this thesis.

Declarations

This thesis is presented in accordance with the regulations for the degree of Doctor of Philosophy. It has been composed by myself and has not been submitted in any previous application for any degree. The work in this thesis has been undertaken by myself except where otherwise stated. All quoted text remains the copyright of the original attributed author. I have taken the liberty of correcting typographical errors, spelling and grammar where necessary.

Some of the content inclusion methods in Chapter 3 have been published in [Landia and Anand, 2009] and [Landia et al., 2012]. The high-level concepts of including content into FolkRank and the adaptation of the graph model of Chapter 4 have been published in the form of an extended abstract as part of a doctoral symposium in [Landia, 2012]. Most of the work addressing graph-based ranking presented in Chapter 4 has been submitted for review to the *ACM Transactions on Information Systems* journal.

Abstract

Tag recommendation algorithms aid the social tagging process in many user-driven document indexing applications, such as social bookmarking and publication sharing websites. This thesis gives an overview of existing tag recommendation methods and proposes novel approaches that address the new document problem and the task of ranking tags. The focus is on graph-based methods such as FolkRank that apply weight spreading algorithms to a graph representation of the folksonomy. In order to suggest tags for previously untagged documents, extensions are presented that introduce content into the recommendation process as an additional information source. To address the problem of ranking tags, an in-depth analysis of graph models as well as ranking algorithms is conducted. Implicit assumptions made by the widely-used graph model of the folksonomy are highlighted and an improved model is proposed that captures the characteristics of the social tagging data more accurately. Additionally, issues in the tag rank computation of FolkRank are analysed and an adapted weight spreading approach for social tagging data is presented. Moreover, the applicability of conventional weight spreading methods to data from the social tagging domain is examined in detail. Finally, indications of implicit negative feedback in the data structure of folksonomies are analysed and novel approaches of identifying negative relationships are presented. By exploiting the three-dimensional characteristics of social tagging data the proposed metrics are based on stronger evidence and provide reliable measures of negative feedback.

Including content into the tag recommendation process leads to a significant increase in recommendation accuracy on real-world datasets. The proposed adaptations to graph models and ranking algorithms result in more accurate and computationally less expensive recommenders. Moreover, new insights into the fundamental characteristics of social tagging data are revealed and a novel data interpretation that takes negative feedback into account is proposed.

Chapter 1

Introduction

Social bookmarking websites such as CiteULike[1], Bibsonomy[2] and Delicious[3] allow users to manage their bookmarks online and share them with others. The methodology used on these sites for indexing and organising the bookmarks is "tagging", which is the process of assigning descriptive or otherwise categorising words to the bookmark. Each user's collection of bookmarks is organised by tags which the user has chosen and the annotated bookmarks are shared publicly, allowing users to browse and search through their peers' bookmarks and tags. Tags can be defined as (key-)words which are assigned to an object in order to describe and index it. The tagged object can be any entity, for example a document, picture, or video clip. The aim of tagging is to group and organise objects and make it easier to find a particular object in the collection. In contrast to a hierarchical organisation, tags are usually not organised in a fixed taxonomy. This unstructured form makes it easier for users to select tags for objects without having to worry about the location of the tags they use in a hierarchy. Additionally, most tagging systems allow users to create their own tags by entering any combination of characters, making the tagging process even more convenient by removing the task of looking through and selecting words from a long list of existing tags. This motivates users to use the system much more heavily than traditional indexing methods based on a predefined structure of categories and labels. The heavy participation of users leads to an increased growth rate of information in social tagging systems compared to traditional indexing solutions. However, the cost of giving so much freedom to the users is that the collaborative aspect of the system suffers in quality. The resulting tags assigned to items in the system are based heavily on the personal scope and

[1] http://www.citeulike.org/
[2] http://www.bibsonomy.org/
[3] http://delicious.com/

choice of individual users. Different users often assign different tags to the same object [Wetzker et al., 2010], and many new tags are introduced by users that are synonyms or variations of existing tags in the system describing the same underlying concept [Gemmell et al., 2008]. This makes each tag assignment more personal and less useful for other users. To preserve the collaborative value of tagging systems, automatic tag suggestion algorithms are used to present the user with a list of existing tags that might be appropriate for the item that he or she is in the process of tagging. The utility of automatic tag recommendation is thus twofold. Firstly, the tagging process is made easier for the individual user by displaying the collective knowledge of the system about the item in question and thus giving hints about how the item could be tagged. Secondly, more homogeneity is introduced into the tag-based organisation of items. If the user accepts one of the suggested tags instead of making up a new label (that could be a synonym or variation of an existing tag), this provides the system with collaborative feedback about the existing tag and the item in question. By choosing existing tags instead of making up their own variations, users help in building and strengthening relationships between items and tags based on collective knowledge. Tag recommendation algorithms thus help to consolidate the users' personal vocabularies to some extent. They help to reduce data sparsity and have a positive effect on the collaborative usefulness of the system.

Users in social tagging systems have different interests in the same items and reflect their personal view in the tags they choose to assign. [Dattolo et al., 2010] carried out an analysis of the different types of tags encountered in social bookmarking systems where the tagged items are textual documents. They conclude that tags are used to describe many different aspects of documents:

- Content: summarising the content of the document
- Type: giving the file type ("pdf", "doc") or publication form ("paper", "blog")
- Related People, Events and Objects: the author of a document, the name of a conference or the name of a related location
- Personal Categorisations and Opinions: "toread", "work" or "interesting" which are only useful to the person who assigned them

Analysing social tagging from the user side, [Körner et al., 2010] examine differences in motivation and tagging behaviour of users in social tagging systems and identify two groups: *categorisers* and *describers*. Categorisers view their tag vocabulary as an organised taxonomy of labels and assign tags to items in order to group and categorise them. Describers use a wider, dynamically changing tag vocabulary to

describe the content of the tagged items. Additionally, [Calefato et al., 2007] highlight that users of social tagging systems have varying levels of expertise regarding the items they tag and thus are likely to use different tag vocabularies. Personalisation of tag suggestions is thus an important aspect in tag recommendation. The tags that are suggested should not only be related to the item to be tagged but also personalised to the tagging preferences of the user. The user is the "customer" of tag suggestion algorithms and the success of recommendations is measured based on whether or not the user accepts them. Nevertheless, apart from the immediate utility to the user, tag recommenders also have a wider-reaching importance since they influence and steer the choice of tags used in the system. From an information retrieval point of view, the tagging of items is part of the index building process of the social bookmarking system. The created associations between users, items and tags are then used as the information base in search applications such as finding items related to a tag or recommending items to users based on their tagging profile. By using tag recommenders, the creation of a denser and more convergent information base (or index) is encouraged. The denser dataset can then be used in search applications to provide better results and recommendations.

1.1 Problem Definition

The work presented in this thesis concentrates on tag recommendation algorithms for social bookmarking systems where the indexed items are textual objects such as websites or research publications, hereafter referred to as documents. The tag recommendation task is to recommend a set of tags to a query user u_q for a query document d_q that the user is in the process of tagging. Each suggested tag that is accepted by the user is a correct recommendation, and each tag that was suggested but not chosen is a wrong recommendation.

As the structure and characteristics of social tagging data is central to the presented work we formally define the social tagging dataset at this point. The data contained in social tagging systems is often referred to as a folksonomy [Jäschke et al., 2007]. A folksonomy is a tuple (U, D, T, A) where U is the set of users, D is the set of documents, T is the set of tags and $A \subseteq U \times D \times T$ is the set of tag assignments. A tag assignment $a \in A$ is a triplet (u, d, t) that indicates that user u has assigned tag t to document d. The folksonomy can be modelled as an undirected tripartite hyper-graph $H = (V, E)$ consisting of nodes $V = U \cup D \cup T$ and hyper-edges $E = \{e_{(u,d,t)} | (u, d, t) \in A\}$ where $e_{(u,d,t)}$ is the hyper-edge modelling the tag assignment (u, d, t) and connecting the nodes representing user u, document d

and tag t in the graph. A post in the folksonomy can be defined as a triplet u, d, S consisting of a user u, a document d and a set of tags $S \in 2^T$, where 2^T denotes the power set of T i.e. the set of all possible tag sets. The set of posts can then be defined as $P \subseteq U \times D \times 2^T$. We use the notation $(u_q, d_q, \emptyset)$ to denote a query post, where u_q is the query user, d_q is the query document and the set of tags is unknown and to be predicted.

1.1.1 Challenges

The challenges in generating successful tag recommendations include the personalisation aspect, the dimensionality and sparsity of tagging data, and the new document problem. These are all general problems of traditional recommender systems, however, they are highly pronounced in the social tagging domain and the tag recommendation task. Due to the mentioned differences in the tagging behaviour of individual users (motivation and expertise), tag recommendations have to be personalised to the preferences of the query user in order to be successfully accepted. The information contained in social tagging data has a high dimensionality in types of objects. In contrast to traditional recommender systems that deal with ratings given to documents by users (such as a user giving a 5-star rating to a book on Amazon[4]), tag recommendation models have to be learnt on data that contains the added dimension of tags. While numerical ratings of the same document by two different users can be compared directly, different tags assigned to the same document cannot be directly compared. The tags are an additional dimension that has to be considered by the models. The additional tag dimension combined with the differences in the tagging behaviour of users leads to a high data sparsity. Furthermore, the query always includes two objects, the user and document, and the task is to recommend tags that are appropriate for these two objects in combination. Similar challenges exist in item recommender systems which consider contextual information [Adomavicius and Tuzhilin, 2011], and where additional data dimensions such as author names or related places are analysed. Another challenge is the new document problem, akin to the *new item* and *cold-start* issues in recommender systems [Jannach et al., 2010]. A large proportion of documents in social tagging systems is only tagged by one user [Wetzker et al., 2008], and thus many query documents for which the tag recommender is asked to make predictions have no previous tagging information associated with them. This is an important issue to address since recommending tags for new query documents based solely on the overall tagging profile of the query user is likely to result in a low success rate. The following list

[4] http://www.amazon.com/

summarises the main challenges faced by tag recommendation algorithms.

- Personalisation: The individual preferences of the query user have to be taken into account.
- Dimensionality of Social Tagging Data: Tag recommendation algorithms have to process data that consists of three-dimensional relationships between user, documents and tags.
- Data Sparsity: Due to differences in the tagging behaviour of users and the added dimension of tags, the data on which recommendations can be based is very sparse.
- New Document Problem: A large proportion of query documents are new and have no previous information associated with them in the social tagging data.

1.2 Objectives

The objectives of this work are to analyse and improve upon existing tag recommendation algorithms for social tagging systems. The aspects addressed by this work can be divided into two main categories: the new document problem and the ranking problem. The first is the issue of recommending tags for previously untagged documents. The second aspect is the problem of ranking the existing tags in the system for recommendation, where the challenges include the aforementioned data dimensionality, sparsity, and the need for personalisation. We address the new document problem by considering the content of documents as an additional information source in the tag recommendation process. The problem of ranking tags for recommendation is addressed by conducting an analysis of existing tag ranking algorithms and proposing novel approaches to leverage the social tagging data. The approach of our work is to identify the scope of information considered by different tag ranking methodologies and propose ways to extract as much predictive information as possible from the sparse folksonomy. The focus is on graph ranking algorithms for tag recommendation as these approaches have the potential to analyse wide-reaching connections in the folksonomy graph and thus have a larger information scope than other methods. We evaluate and compare all of our presented approaches on real-world datasets from three popular social tagging websites, and conduct experiments on full unpruned test sets as well as post-core subsets of the data.

1.3 Thesis Structure

The rest of the thesis is structured as follows. The remainder of the introduction summarises our main contributions. In Chapter 2 we give an overview and analysis of existing tag recommendation approaches, evaluation methodologies and datasets. Then we present our contributions, where Chapter 3 addresses the new document problem, and Chapters 4 and 5 address the task of ranking tags for recommendation. Finally, we summarise our conclusions and give suggestions for future work in Chapter 6.

1.4 Contributions

In the following subsections we give an overview of the main contributions of this work, where each contribution corresponds to a chapter in the thesis.

1.4.1 Content-Awareness

In Chapter 3 we address the new document problem in tag recommendation. To be able to recommend tag for previously untagged documents, we propose novel extensions to folksonomy-based recommenders that consider the content of documents as an additional information source. Existing combinations of folksonomy-based and content-based approaches usually construct a hybrid recommender that blends prediction sets generated by individual folksonomy-based and content-based components. We address this hybridisation problem from a different perspective by including content into folksonomy-based approaches at the algorithmic level, before predictions are generated. Our content-aware extensions allow the folksonomy-based algorithms to compare the content of previously untagged documents to content that has been tagged in the past in order to generate recommendations. We propose and discuss two different methods for creating content-aware, folksonomy-based recommenders and apply our methodology to FolkRank [Hotho et al., 2006c], a graph-based ranking algorithm, and also to a less expensive recommender based on co-occurrence. Including content into the recommendation process provides a significant increase in the recommendation accuracy of the algorithms on real-world tagging datasets. Another interesting discovery of this chapter is that our simpler co-occurrence recommenders produce comparable and in some cases better results than the graph-based FolkRank, with including content data as well as without. This is surprising since FolkRank has a much larger information scope and up to now has been thought to be superior to simple ranking approaches such as co-occurrence.

1.4.2 Graph-Based Ranking

Prompted by the discoveries of Chapter 3, we conduct a detailed examination of the inner workings of FolkRank, and analyse issues in the application of graph-based ranking algorithms to the tag recommendation problem in general. We present this work in Chapter 4, where we examine implicit assumptions made by the widely-used graph model of the folksonomy as well as FolkRank's weight spreading algorithm. We highlight how these assumptions can have a negative impact on tag rankings and propose novel adaptations in order to overcome the issues. We present and evaluate an improved graph model that captures the social tagging data more accurately and increases the recommendation accuracy of FolkRank. To address issues in FolkRank's tag rank computation, we propose an adapted weight spreading algorithm for social tagging graphs. Our approach allows for a more comprehensive analysis of the weight spreading process in social tagging data, and results in a recommender that produces comparable results to FolkRank while being computationally much less expensive. Moreover, we present an in-depth theoretical discussion as well as practical evaluation of the benefit of using complex graph ranking algorithms over simpler methods that consider just one level of co-occurrence. Since considering a larger scope of information in tag ranking algorithms comes at a computational expense, we evaluate the impact and benefit of considering each additional level of information with graph ranking methods. We conclude that no significant benefit is gained by analysing deeper levels of folksonomy data with conventional weight spreading approaches and propose that this could be due to the underlying characteristics of the social tagging data. In particular, our results suggest that the base assumption of weight spreading, that closeness in the graph always indicates a positive relationship between nodes, might not hold for folksonomies.

1.4.3 Negative Feedback

In Chapter 5 we expand on our conclusions from the previous chapter and show that some of the multi-hop connections in the folksonomy graph indicate a negative relationship between nodes instead of a positive one. Conventional weight spreading methods are built on the base assumption that closeness in the graph always indicates a positive relationship, where the strength of positive relatedness is inversely proportional to the distance (or number of hops) between two nodes. We show that this assumption does not hold for folksonomy graphs and that some of the relatively short paths between two nodes actually indicate a negative relationship. If these relationships are assumed to be positive, as is done by conventional methods,

they can hinder the algorithms ability to generate correct recommendations. We propose a novel method for identifying and measuring negative relationships in the graph structure of the folksonomy. Our approach exploits the three-dimensional relationships in social tagging data and is based on stronger evidence that existing approaches for extracting negative feedback. We evaluate the impact on the accuracy of tag predictions if negative relationships are misinterpreted as being positive and show that better results can be achieved by correctly identifying the negative feedback present in the graph structure. The existence of negative feedback gives fundamental insights into the characteristics of social tagging data and provides additional predictive information that can be used to design improved tag recommendation algorithms in the future.

Chapter 2

Analysis of Existing Approaches

This chapter contains a survey and analysis of existing tag recommendation approaches and algorithms. The different types of algorithms are examined in detail, followed by a brief description of existing evaluation methodologies and datasets used in research. We conclude the chapter by giving a high-level summary of existing tag recommendation approaches and explaining how the contributions of this work relate to existing research.

Tag recommendation solutions can be categorised into folksonomy-based and content-based approaches. Folksonomy-based approaches model and analyse the relationships between users, documents and tags in the social tagging data in order to generate recommendations. The tag recommendations for a query post are generated by analysing the tagging profiles of the query user and query document. While folksonomy-based approaches achieve a high prediction accuracy, they have a major drawback in that tags can only be recommended for documents that have already been tagged at least once in the historical data (new item problem). For documents that are new, only the query user can be considered by solely folksonomy-based algorithms. In these cases, the recommended tags are personalised to the query user, however, whether or not they are adequate for the query document cannot be taken into account since no information about the query document is available to the algorithms. Content-based approaches consider the textual content and/or meta-data of documents. They build models which reflect the relationship between the document content and the tags assigned to the document. Through utilising content, tags can be recommended that are related to the content of the query document, even if the query document itself has not been tagged before. Content-based approaches originate from the fields of information retrieval and document categorisation and traditionally address an un-personalised recommendation task.

The focus is on recommending tags or labels that globally reflect the content of the document rather than the opinions of individual users about the document. For the personalised tag recommendation task, content-based approaches have to be combined with folksonomy-based approaches in order to take the preferences of the query user into account. This combination can be achieved by using hybrid approaches that blend the tag prediction scores generated by several component recommenders to create the final recommendation set. The individual components of the hybrid can include several folksonomy-based and/or content-based recommenders.

2.1 Folksonomy-Based

Methodologies relying on the folksonomy data include Hypergraph [Symeonidis et al., 2008; Rendle et al., 2009], Graph [Jäschke et al., 2007; Ramezani et al., 2010], collaborative filtering [Gemmell et al., 2009; Xu et al., 2006] and simple co-occurrence [Lipczak and Milios, 2011; Landia et al., 2012] approaches. While hyper-graph approaches try to capture and analyse all characteristics of the folksonomy in their models, regular graph, collaborative filtering and co-occurrence approaches can be described as reductionist methods since they reduce the 3-dimensional folksonomy data to one or several 2-dimensional projections. The benefit of this is that the resulting models are less complex (thus computationally less expensive) and aggregating the data from 3 to 2 dimensions produces a denser dataset which potentially reveals some additional characteristics of the data. However, on the other hand, the information about 3-dimensional inter-relationships between users, documents and tags present in the folksonomy is ignored and lost by using these reductionist models [Symeonidis et al., 2008]. An important difference between graph-based, collaborative filtering and co-occurrence approaches is the information scope considered in the data models, and the size of the candidate tag sets. Co-occurrence models only consider the immediate co-occurrence neighbourhood of objects in the folksonomy. The candidate tag set only includes tags co-occurring with the query, for example tags co-occurring with the query user. Collaborative filtering models consider connections one level deeper into the folksonomy, for example comparing the query user to other users based on their overlap in document sets or tag vocabulary. The candidate tag set then consists of tags co-occurring with the query user as well as tag co-occurring with similar users, and the tagging behaviour of similar users is considered when making tag predictions. Graph-based approaches have the possibility to analyse deeper connections in the tagging data and include information contained in the full folksonomy in their models. The candidate tag

set potentially includes all tags in the historical data. Graph-based methods have the ability to make predictions about and rank all existing tags in the system with regard to a query.

2.1.1 Co-Occurrence

Approaches based on co-occurrence are the simplest of the tag recommendation methods. The set of candidate tags in theses approaches consist only of the tags co-occurring with the query user or query document in the historical data. The score of each candidate tag is calculated based on its co-occurrence strength with the query user u_q or the query document d_q. To finally recommend tags that are related to the document as well as personalised to the user, the user-related and document-related co-occurrence scores are usually combined. In user-tag co-occurrence, the score of each tag t is given by the number of times that the query user u_q has used the tag in the past. Formally, the user-tag co-occurrence count, utCount, is defined as

$$\text{utCount}(u_q, t) = |\{(u, d, t') \mid (u, d, t') \in A \wedge u = u_q \wedge t' = t\}|$$

where A is the set of tag assignments. The co-occurrence count thus measures the number of tag assignments containing the query user $u_q \in U$, the tag $t \in T$ and any document $d \in D$. For simplicity, we use the notation $(u, d_\exists, t)$ to denote tag assignments where the user is a specific user $u \in U$, the tag is a specific tag $t \in T$ and the document $d_\exists \in D$ is any document in the collection. We then write the user-tag co-occurrence count as

$$\text{utCount}(u_q, t) = |(u_q, d_\exists, t) \in A|$$

Similarly, document-tag co-occurrence count is given by

$$\text{dtCount}(d_q, t) = |(u_\exists, d_q, t) \in A|$$

where d_q is the query document and $u_\exists \in U$ is any user in the folksonomy. In [Jäschke et al., 2008], the raw user-tag and document-tag counts are used directly to recommend tags. An improved co-occurrence measure, which we refer to as co-occurrence probability, is proposed by [Lipczak et al., 2009]. The user-tag co-occurrence probability (utProb) considers not only the raw co-occurrence count of the u_q and t but also takes the total number of posts of the query user into account:

$$\text{utProb}(u_q, t) = \frac{|(u_q, d_\exists, t) \in A|}{|(u_q, d_\exists, S_\exists) \in P|}$$

where the numerator is the co-occurrence count (utCount) and the denominator is the total number of posts made by u_q, tagging any document $d_\exists$ with any set of tags $S_\exists$. Thus, utProb(u_q, t) represents the probability that user u_q will use tag t for any post. Similarly the document-tag probability is given by

$$\text{dtProb}(d_q, t) = \frac{|(u_\exists, d_q, t) \in A|}{|(u_\exists, d_q, S_\exists) \in P|}$$

where the numerator is equal to dtCount and the denominator is the number of total posts concerning the query document d_q. When considering only one side of the query, for example the user-related co-occurrence on its own, there is no difference in the ranking order of the recommended tags in utCount and utProb. Since the total number of posts is constant for one user, the tag scores in utProb are those of utCount divided by a constant. However, the difference between the raw count and the probability measures becomes important when the user-related and document-related co-occurrence scores are combined.

2.1.2 Collaborative Filtering

The general idea behind Collaborative Filtering (CF) is that information generated by all users in a community can be used to filter and select objects to be recommended to one individual user. CF originates from the field of recommender systems [Jannach et al., 2010] and is designed to deal with ratings given to items by users [Gemmell et al., 2009]. However, in the tag recommendation context, instead of numerical ratings the user assigns tags to items. While all numerical ratings can be compared across assignments, tag assignments cannot be compared directly across different tags. This adds another dimension to the data which is not present in recommender systems. In recommender systems the data can be represented as a 2-dimensional matrix of users and items $U \times I$ where each entry in the matrix is the rating given to item i by user u. With tags, the adjacency tensor $C = U \times D \times T$ of the folksonomy data has 3 dimensions. The data has to be either projected onto two dimensions [Jäschke et al., 2007; Zanardi and Capra, 2008; Gemmell et al., 2009] in order to process it with classic collaborative filtering algorithms, or the collaborative filtering approaches have to be adapted to handle the added tag dimension [Gemmell et al., 2009; Xu et al., 2006; Lu et al., 2011].

Classic CF

Traditional collaborative filtering can be divided into user-based and item-based CF. User-based CF relies on the theory that each user belongs to a group of similarly

behaving users. In the context of tag recommendation, user-based CF calculates tag scores based on the usage of tags by users with a similar profile. In order to construct a vector representation of users and be able to calculate the similarity between them, a 2-dimensional matrix is extracted from the 3-dimensional folksonomy adjacency tensor $C = U \times D \times T$. Users can alternatively be described by the documents they have tagged by constructing $UD = U \times D$ or by the tag sets they have used via $UT = U \times T$. [Jäschke et al., 2007] use binary matrices UD and UT. For UD, only binary information is available since each user can only tag each document once with a set of tags. For UT however, a non-binary matrix can also be used that takes into account how many documents the same tag was assigned to by a user [Zanardi and Capra, 2008]. Users can then be represented as the row vectors of one of the two alternative matrices and the similarity between users can be calculated by cosine similarity. Using the similarity between users, a neighbourhood of the k most similar users is constructed for the query user u_q. The set of candidate tags for recommendation to u_q includes all tags associated with users from the k-neighbourhood of u_q. The score for each candidate tag t is then calculated based on the similarity to the query user of all users who have used t. If a non-binary matrix UT is used, the strength of the relationship between each user in the neighbourhood and each candidate tag t can also be taken into account. Using the UT matrix, a possible formula to calculate the final tag score for each candidate tag t by user-based collaborative filtering (UCF) is

$$\mathrm{UCF}(u_q, t) = \sum_{u \in N(u_q)} \mathrm{sim}(u_q, u) * UT_{u,t}$$

where $N(u_q)$ is the neighbourhood of k users most similar to the query user u_q, $\mathrm{sim}(u_q, u)$ is the similarity between u_q and user u from the neighbourhood, and $UT_{u,t}$ is the entry for user u and tag t in UT. User-based CF focuses entirely on the user side of the query, ignoring the query document. Since in tag recommendation the task is to recommend tags to a user for a specific document, it is common to filter the recommendations generated by user-based CF to include only tags that have been assigned to the query document in the past [Jäschke et al., 2007; Gemmell et al., 2009]. If this filtering is done, the complete approach can be described as taking the past tags of the query document as the candidate tag set, and ranking these tags according to the preferences of the query user and other users with similar interests.

Item-based CF [Deshpande and Karypis, 2004] works in a similar fashion to user-based CF, by finding the neighbourhood of items similar to the query item, where in our case the items are documents. Tag scores are then calculated based

on the frequency with which each tag appears in the document's neighbourhood of similar documents. Documents can be represented as vectors of users by constructing the binary matrix $DU = D \times U$ or vectors of tags by constructing $DT = D \times T$ which can either be binary or include the co-occurrence strength of documents and tags. Byde et al. use item-based collaborative filtering for tag recommendation in [Byde et al., 2007], representing documents as row-vectors of a non-binary DT matrix.

Adapted CF

In [Gemmell et al., 2009], an adapted k-Nearest Neighbour (k-NN) algorithm for tag recommendation is proposed. Instead of representing users by either their document sets or their tag sets, an improved user model is presented which incorporates both the user's tag set and user's document set at the same time. Additionally, information about which tag was assigned to which document by the user is also included. Recommending tags with the adapted *k-NN* algorithm is based on finding a neighbourhood of users similar to the query user. However, as with classic user-based CF applied to tag recommendation, from the tags appearing in the neighbourhood of the query user u_q, only those are considered which have been assigned to the query document d_q in the past. The score for each of these candidate tags t is then calculated by the average similarity of the neighbours of u_q who have assigned tag t to document d_q.

In [Xu et al., 2006] Xu et al describe a collaborative tag suggestion system based on user authority. The authority score for each user is calculated by the average quality of the tag assignments submitted by the user. It is given by

$$\text{Auth}(u) = \sum_{d_j \in D(u)} \frac{\sum_{t_k \in T(u,d_j)} \text{Strength}(d_j, t_k)}{|T(u, d_j)|}$$

where $D(u)$ is the set of documents tagged by user u, $T(u, d_j)$ denotes the set of tags assigned to document d_j by u, and $\text{Strength}(d_j, t_k)$ is the goodness measures of the tag assignment. The goodness measure captures the general strength of the relationship between a document and tag. It is independent of individual users, but it is based on the collective authority of all users who have made the corresponding tag assignment. $\text{Strength}(d, t)$ is calculated as the sum of the authority scores of all users who have assigned t to d:

$$\text{Strength}(d, t) = \sum_{u_i \in U(d,t)} \text{Auth}(u_i)$$

Auth(u) and Strength(d, t) are computed by an iterative algorithm similar to HITS [Kleinberg, 1999], where initially the authority of each user is set to the same value of one.

2.1.3 Tag Expansion

While most approaches model relationships of the form user-tag and document-tag, tag-tag relatedness has also been explored as a means to improve the quality of the recommended tag set. While user-tag and document-tag models try to identify individual tags that have a high chance to be accepted, the idea behind using tag-tag relatedness measures is that the recommendation set is seen as a whole. Thus, the focus is on recommending a collection of tags that together are appropriate for the query post. This methodology is applied to the traditional tag recommendation task, where only the query user and query document are given, by generating a small initial recommendation set of tags and then expanding it by adding further tags through tag-tag relatedness [Lipczak et al., 2009; Xu et al., 2006]. While [Lipczak et al., 2009] include all tags from the initial recommendation set in the tag expansion process, [Xu et al., 2006] only use the highest scoring initial tag for tag expansion and ignore the rest. An assumption made by these approaches, especially in [Xu et al., 2006], is that the system has a high confidence that the topic area or tag cluster identified by the limited number of initial tags will be correct and thus produce further correct tags via tag expansion.

Widely-used approaches for calculating tag-tag relatedness include co-occurrence [Lipczak et al., 2009; Sigurbjörnsson and van Zwol, 2008] and vector similarity approaches [Zanardi and Capra, 2008]. In [Cattuto et al., 2008] Cattuto et al. evaluate the usefulness of different measures by comparing the values given by the tag-tag relatedness metrics to the distance of the words in WordNet [Fellbaum, 1998]. A post-based co-occurrence measure, three distributional measures and a weight-spreading measure (implemented by FolkRank [Hotho et al., 2006a]) are examined. The post-based co-occurrence between two tags t_k and t_l is defined as the number of posts which contain both t_k and t_l. The evaluated distributional measures are based on three alternative vector space representations for tags. A tag t_k is either represented by a vector of the number of times t_k is assigned to each resources in the system, a vector of the number of times each user has used t_k, or a vector of t_k's post-based co-occurrence with the other tags in the system. The tag-tag relatedness based on weight-spreading is calculated by setting an appropriate preference vector for FolkRank and performing weight spreading iterations (see Section 2.1.4). Another possible approach to creating tag-tag relationships is by

organising tags into clusters [Gemmell et al., 2008; Shepitsen et al., 2008].

2.1.4 Graph-Based

Graph-based methods combine concepts from all of the previously described approaches into one data model. Starting from the query user and query document, they analyse multi-hop connections in the folksonomy hyper-graph, going beyond the immediate neighbourhood of the query nodes. By doing this deep analysis of the folksonomy data, graph-based approaches combine ideas from co-occurrence, collaborative filtering, as well as tag-tag relatedness approaches into one algorithm. Graph-based algorithms consider the complete folksonomy when generating predictions for a query document and user, and thus have a much larger information scope than the previously mentioned methods. The candidate tag set in graph-based approaches potentially includes all of the existing tags in the folksonomy if the whole graph is connected.

Hyper-Graph

Hyper-graph approaches directly follow the folksonomy definition and model the data as a hyper-graph $H = (V, E)$ consisting of nodes $V = U \cup D \cup T$ and hyper-edges $E = \{e_{(u,d,t)} | (u, d, t) \in A\}$. [Symeonidis et al., 2008] construct the corresponding adjacency tensor of nodes V as a 3-dimensional binary matrix, $C = U \times D \times T$ where each entry $c_{u,d,t} \in \{0, 1\}$ specifies whether or not $(u, d, t) \in A$. To recommend tags, a lower-rank approximation of C is created which preserves the 3-dimensional relationships between users, documents and tags. The approximation is created by applying tensor factorisation techniques to split C into several components, and then reconstructing a denser version of the adjacency tensor, C'. Each entry $c'_{u,d,t}$ in C' is a decimal value that represents the predicted probability that user u will assign tag t to document d, and C' is then used to generate tag recommendations for a query post $(u_q, d_q, \emptyset)$. To create C', Symeonidis et al. use High Order Singular Value Decomposition (HOSVD) on the binary adjacency tensor C. [Rendle et al., 2009] propose an alternative factorization approach specifically for folksonomy datasets called Ranking Tensor Factorisation (RTF) that can deal with missing values in C. An important difference between the two approaches lies in the interpretation of the hyper-graph and the construction of the initial adjacency tensor C. While Symeonidis et al. set all entries $c_{u,d,t} \in C$ for which $(u, d, t) \notin A$ equal to zero, Rendle et al. argue that some of the non-occurring tag assignments should be treated as missing values instead. In their approach, the entries for $c_{u,d,t} \in C$ where user u has

not tagged document d with any tags are set to be unknown, and only the entries for which there exists a post $(u, d, S) \in P$ but $t \notin S$ are set to indicate a low ranking of t for u and d. Further factorisation techniques using this missing values approach are presented in [Rendle and Schmidt-Thieme, 2010].

Graph

Graph approaches relax some of the constraints of the folksonomy hyper-graph, and model the data as a tri-partite or several bi-partite graphs, where nodes are connected with normal edges instead of hyper-edges. While the hyper-edges in the hyper-graph are "planes" that always connect three nodes (u, d, t), edges in the graph model are "lines" which connect only two nodes: either (u, d) or (u, t) or (d, t).

FolkRank The most prominent graph-based approach for social tagging data is FolkRank [Hotho et al., 2006c; Jäschke et al., 2007], a ranking algorithm modelled after Google's PageRank [Brin and Page, 1998; Bianchini et al., 2005]. Similarly to PageRank, the key idea of FolkRank is that a document which is tagged by important users with important tags becomes important itself. The same holds symmetrically for users and tags. Users, documents and tags are represented as nodes $v \in V$ in an undirected tri-partite graph $G = (V, E)$, where all co-occurrences of users and documents, users and tags, and documents and tags are edges $e \in E$ between the corresponding nodes. The weight of the edge between two nodes depends on the number of their co-occurrences, measured by the number of tag assignments that both nodes appear in.

The importance or rank of each node in FolkRank is calculated by an iterative weight-spreading algorithm in a similar fashion to PageRank. The weights of all nodes are given in the weight vector $\vec{w}$ which has one entry per node and is computed by the weight spreading function

$$\vec{w} \leftarrow (1 - d)M\vec{w} + d\vec{p}$$

where M is the row-stochastic version of the adjacency matrix of graph G, $\vec{p}$ is the preference vector, and the dampening factor $0 < d \leq 1$ determines the influence of $\vec{p}$. The preference vector $\vec{p}$ is used as a means to personalise the recommendations to the query user and document, and to achieve that goal is set to give the nodes representing the query user and document in the graph a higher preference weight compared to other nodes. The dampening factor d sets the balance between personal

preference and global importance when calculating the node weights. FolkRank can generate either a global non-personalised ranking or a personalised ranking of all nodes in the graph, depending on the values set in the preference vector. For a global ranking, the entries in $\vec{p}$ for all nodes are set to the same value. In order to generate personalised recommendations for a query post $(u_q, d_q, \emptyset)$, $\vec{p}$ is set so that higher preference weights are given to the query user u_q and query document d_q, compared to other nodes in the graph which are set to have a uniformly small preference weight [Hotho et al., 2006a]. As our work is focused heavily on graph-based approaches, FolkRank is analysed and discussed in detail in Chapters 3 and 4.

TriFolkRank In [Kim and El Saddik, 2011], Kim et al. further explore the FolkRank algorithm and suggest an adapted approach called TriFolkRank. TriFolkRank calculates two FolkRank vectors individually, one for the query user by setting a user-only preference vector, and one for the query document by setting a document-only preference vector. A linear combination of the two FolkRank vectors then gives the final ranks for tags. The benefit of this approach is that the individual tag ranking vectors for each existing user and document in the folksonomy can be pre-calculated offline. During the online tag recommendation phase the already computed tag rang vectors of the query user and query document can be retrieved and combined to produce the final recommendations. The expensive weight spreading computation is thus moved to an offline pre-calculation phase and the speed of the online generation of tag recommendations is significantly improved.

Directed Graph A directed graph alternative to the undirected folksonomy graph is suggested by Ramezani et al. [Ramezani et al., 2010; Ramezani, 2011]. The rationale behind using directed edges is explained by considering the expectation of hypothetical user navigation from one node to another. Ramezani et al. argue that if we consider two connected nodes in the graph, out of which one is popular and the other unpopular, navigating from the unpopular node to the popular one is different from and more likely than navigating from the popular node to the unpopular one. In other words, when navigating from a node v_1 to a node v_2, not only the strength of the connection between v_1 and v_2 is important, but also the overall popularity/likeliness of v_2. Since v_1 and v_2 can have different levels of popularity, going from v_1 to v_2 can have a different probability than going from v_2 to v_1. To be able to capture this information in the graph, a directed graph with adapted edge weights is used.

BiFolkRank Another alternative graph method proposed in [Kim and El Saddik, 2011] is BiFolkRank. In BiFolkRank, the $U \times D \times T$ tri-partite graph used by FolkRank is split into two bi-partite graphs $U \times T$ and $D \times T$. The weight spreading algorithm is then executed on either $U \times T$ or $D \times T$, ignoring either documents or users respectively. In each case, the resulting tag rankings are then combined with the tags that co-occur with either the ignored query document or the ignored query user to give the final recommendations. Surprisingly, both of these simpler approaches are shown to give better results than the original FolkRank approach. Even though this seem unintuitive at first, the conclusions of the work presented in this thesis provide an explanation as to why this might be the case.

2.2 Content-Based

The focus of content-based approaches is to generate content-related tags which describe the document from a neutral point of view. By utilising the content of documents as an information source, these approaches have the ability to recommend tags for previously untagged documents and can address the new document problem in tag recommendation. Content-based approaches usually address an un-personalised tag recommendation task. The datasets used in traditional document labelling approaches consist of documents globally annotated with keywords that reflect the content of the document, and have usually been assigned by experts in the field. However, in social tagging data users have varying expertise and assign content-related as well as personal tags that represent the user's personal view of the document. When content-based approaches are used on social tagging datasets, the models are usually built on training data that is pre-processed by removing the user dimension. Optionally the un-personalised training data can be further reduced by keeping only the most frequently assigned tags per document in order to reduce noise and create a denser dataset. To finally recommend personalised tags for a document to a specific user, the initial list of content-related tags is usually adjusted in a second step according to the preferences of the user.

Content-based approaches use the text content of documents to either directly extract tags [Lipczak et al., 2009] or to create a content-based document model in order to recommend tags. Approaches based on document models include traditional document classification algorithms [Heymann et al., 2008; Song et al., 2008] and approaches based on document similarity [Byde et al., 2007]. Important aspects of content-based methods are the content source and the document representation. Various sources of content data such as the title, additional meta-data

and fulltext content of documents are available. The document representation used is usually a bag-of-words, however, there are different approaches to calculate the weight of each word in a document.

2.2.1 Content Sources

The available sources of content data for documents in social bookmarking systems include the title, URL, additional meta-data and fulltext content. Experiments have shown that the most informative words generally appear in the title and URL, and the document text. An extensive analysis of the usefulness of title words for tag recommendation is given by Lipczak et al. in [Lipczak and Milios, 2010b]. The conclusion is that there is a relatively high overlap between the title words and tags of a document, and the title seems to be a good source for words to be used as tags. The value of the document URL, as a source of potential tags or as a document representation, has been explored in [Lipczak et al., 2009] and [Byde et al., 2007]. Lipczak et al. conclude that while the URL-based tag recommender on its own does not achieve very good results, it is valuable in combination with other tag recommendation sources. If provided by the tagging system, additional meta-data fields such as *description*, *author* or *journal* can also be analysed [Ju and Hwang, 2009; Landia et al., 2012]. For structured text documents such as HTML, the content structure (anchors, links, paragraphs) can be exploited and different areas of the document can be analysed as separate content sources [Zhang et al., 2004]. In [Heymann et al., 2008] Heymann et al. carry out experiments on HTML pages, comparing the value of page text, anchor text and text of surrounding hosts for tag prediction. They conclude that out of the three, the document text was most informative and anchor text was more informative than surrounding hosts.

2.2.2 Content Representation

The document representation in content-based approaches is usually a bag-of-words. Each document is represented as a vector of word importance weights where each element of the vector represents the importance of a word to the document. The standard word scoring approach is Tf-Idf which stands for *Term frequency - Inverse document frequency* and measures the ability of a word to distinguish a document from other documents in the collection. Many variations of the Tf-Idf score have been proposed and used in research. One of the standard methods to calculate Tf-Idf is

$$\text{Tf-Idf}(w,d) = \frac{\text{tc}(w,d)}{\sum_{w_l \in d} \text{tc}(w_l,d)} * \log_2 \frac{|D|}{\text{dc}(w,D)}$$

where D is the set of all documents in the collection, $\mathrm{tc}(w, d)$ is the term count equal to the number of occurrences of word w in document d, and $\mathrm{dc}(w_l, D)$ is the document count equal to the number of documents in the collection containing word w. The left side of the multiplication is the term frequency measuring the fraction of occurrences of word w over occurrences of all words in d. The right side is the $\log_2$ of the inverse document frequency. The document frequency measures the fraction of documents in collection D that contain word w.

Additionally to Tf-Idf, different approaches have been used in combination to construct the final document weight vectors. [Witten et al., 1999] use Tf-Idf scores as well as the position of the first appearance of a word in the document; [Hulth, 2003] combines Tf-Idf scores and lexical tools; and [Liu et al., 2009] use Tf-Idf scores with part-of-speech analysis, word clustering and a sentence importance score. In [Renz et al., 2003] the Tf-Idf scores are calculated on small word parts (quad-grams consisting of 4 letters) instead of whole words to overcome the stemming [Porter, 1980] problem. Alternatively to Tf-Idf, [Matsuo and Ishizuka, 2004] use word frequency and word-word co-occurence to calculate scores for words given only a single document rather than a document collection.

An issue in modelling content is the high dimensionality of the document vectors, especially if the full text of the documents is used as the content source. In order to reduce the dimensionality, feature selection can be applied to restrict the document vectors to include only a number of the most valuable words. Various metrics for measuring the value of words for feature selection are compared and evaluated in [Yang and Pedersen, 1997]. Yang and Pedersen conclude that thresholding words based on document frequency is a cheap and effective approach, producing comparable results to more expensive metrics such as information gain. A more advanced approach to dimensionality reduction is feature extraction, where the document word vectors are mapped onto a number of latent features extracted from the document collection. Feature extraction approaches include latent semantic analysis [Deerwester et al., 1990] and probabilistic latent semantic analysis [Hofmann, 1999].

2.2.3 Classification

Classification approaches treat the tag recommendation problem as a multi-label document classification task, where the text content of documents are the features and the tags are labels. They construct a classifier from a training set of labelled documents and apply it to new documents to find keywords. A good general overview of text classification algorithms is given in [Sebastiani, 2002].

In [Heymann et al., 2008] Heymann et al. apply classification techniques util-

ising implicit negative examples in order to recommend un-personalised tags for documents. The dataset used in the experiments only includes documents that have been tagged in more than 100 posts. In order to generate a training set of positive as well as negative examples for classifiers, two sets of relationships between tags and documents are constructed:

- R_p contains pairs (t, d) where tag t positively describes document d.
- R_n contains pairs (t, d) where tag t negatively describes document d.

A tag-document pair (t, d) is included in the set of positive examples R_p if the number of co-occurrences of t and d is greater or equal to the number of total posts for d divided by 100. The set of negative examples R_n includes all pairs (t, d) for which there are no co-occurrences of t and d. Tag recommendations are then generated by training a binary Support Vector Machine (SVM) classifier for each tag, on the content-based representations of the positive and negative example documents in R_p and R_n.

An un-personalised tag recommender based on clustering and classification is suggested by Song et al. [Song et al., 2008, 2011]. The relationship between documents, document vocabulary and document tags is modelled with two bipartite graphs, one giving the relation between the word vocabulary of the dataset and individual documents, the other capturing the connections between documents and associated tags. The two bipartite graphs are approximated by a lower dimensional representation and partitioned into a predefined number of clusters. For each tag a within-cluster ranking is calculated. A classifier is then built treating the clusters as classes and using the distribution of documents and words among classes as features. To recommend tags for a new document, the cluster-membership probability of the new document to each cluster is first predicted by the classifier. The final score for each tag is then calculated by considering the membership probability of the new document to the tag's cluster and the within-cluster rank of the tag.

2.2.4 Document Similarity

The content-based similarity between documents can be used to create a neighbourhood of documents with similar content to the query document. Tags co-occurring with documents in the neighbourhood can then be recommended for the query document [Byde et al., 2007; Landia and Anand, 2009]. Document similarity approaches are similar to item-based collaborative filtering in that the neighbourhood of similar documents is used to recommend tags. However, the neighbourhood is constructed

based on content similarity rather than tag profile similarity. The content-based similarity is usually calculated by cosine similarity of the bag-of-words vectors representing documents. The cosine similarity of two attribute vectors $\vec{x}$ and $\vec{y}$ is calculated by

$$\text{CosSim}(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\|\|\vec{y}\|} = \frac{\sum_{i=1}^{n} \vec{x}_i \times \vec{y}_i}{\sum_{i=1}^{n} (\vec{x}_i)^2 \times \sum_{i=1}^{n} (\vec{y}_i)^2}$$

where the numerator is the dot product of $\vec{x}$ and $\vec{y}$, and the denominator is the product of the magnitudes of $\vec{x}$ and $\vec{x}$. We discuss our tag recommendation approach based on document similarity in more detail in Chapter 3.

2.2.5 Keyword Extraction

A research area related to tag recommendation is keyword extraction, which is the task of extracting keywords from the text content of a document which describe or categorise the document from a neutral point of view. The distinguishing characteristic of keyword extraction in contrast other content-based document labelling methods is that the vocabulary of potential keywords consists of the content of the documents. For the tag recommendation task, keyword extraction is used to identify words appearing in the content of documents that can be recommended as tags. Keyword extraction methods thus cannot recommend tags that are not found as words in the content vocabulary of the document collection, even if the tags exist in the folksonomy tagging data. However, keyword extraction has the ability to generate and recommend new tags that do not yet exist in the tagging data from the document vocabulary, addressing the new tag problem.

Existing approaches can be divided into supervised and unsupervised keyword extraction. Unsupervised methods [Liu et al., 2009] rely solely on the text content of the document collection to identify keywords that provide an accurate representation of the documents. The true document labels are used only as a means to evaluate the accuracy of the keywords extraction approaches and are not analysed as an information source by the algorithms. Unsupervised keyword extraction is closely related to document content representation methods, as the tagging information is not taken into account when generating the set of keywords for a document. Similarly to content representation metrics, the problem addressed by these approaches is to identify (key-)words in the content of a document which accurately represent the document in the collection. Unsupervised methods calculate a score for each candidate word in a document to measure how useful the word would be as a keyword and recommend the words with the highest scores. The score calculation

usually includes Tf-Idf in combination with other metrics, as described in Section 2.2.2.

Supervised approaches learn from the document content as well as the keywords assigned to documents. In order to extract keywords they build models on training data which is assumed to be a collection of correctly tagged documents. In contrast to models for tag recommendation or document classification that try to predict labels for documents, supervised keyword extraction builds models to predict whether or not a word in the content of the document collection should be used as a label. KEA [Witten et al., 1999] uses uses a Naive Bayes classifier to extract keywords and [Zhang et al., 2004] propose a supervised method specifically for web sites using the C5.0 decision tree classifier. A keyword extraction approach that has been directly applied to tag recommendation [Lipczak et al., 2009; Lipczak and Milios, 2011] is discussed below.

Word-Tag Overlap A supervised keyword extraction approach that is applied to tag recommendation with a high success rate is presented in [Lipczak and Milios, 2011] as the "Title Recommender". It recommends words from the query document's title, choosing the words which have been observed to have a high global overlap in being a title word as well as a tag for documents. The score used by Lipczak et al. represents the global suitability of a word to be tag. It is calculated as

$$\text{suit}(w) = \frac{|w \in d \wedge w \in T(d)|}{|w \in d|}$$

where w is a word, d is a document and $T(d)$ is the set of all tags assigned to document d (by any user). The numerator gives the number of documents for which w is a content word as well as a tag, and the denominator is the total number of documents for which w is a content word. The suitability score does not include a word importance score such as Tf-Idf, thus the varying importance of different content words to a document is not taken into account. However, as the only source of content words in [Lipczak and Milios, 2011] is the document title, it can be assumed that all title words are important to the document. The recommender thus achieves a high precision without considering this aspect but can only recommend a small number of tags.

2.3 Other Approaches

Other existing approaches include analysing the time aspect of tagging activity by considering recency [Lipczak and Milios, 2010a], and including information from external sources such as online repositories [Mrosek et al., 2009]. Recency is especially interesting since the timestamps of posts are a data feature present in the folksonomy which is not considered by most approaches. As [Hotho et al., 2006b] point out, the interest of users in folksonomies changes over time, and thus [Lipczak and Milios, 2010a] argue that tags used in the past might not accurately reflect the current interest of a user. As part of their hybrid, Lipczak et al. propose a user-related tag recommender that takes recency into account. As with user-tag co-occurrence, the candidate set of tags consists of all tags used by the query user u_q in the past. However, rather than scoring each tag t based on the number of times it was used, the tag score is calculated based on the recency of the last tag assignment of u_q which contained t.

2.4 Hybrids

Hybrid approaches [Lipczak and Milios, 2010a; Gemmell et al., 2010; Zhang et al., 2009] combine the predictions generated by several individual recommenders in order to create the final recommendation set. Similarly to classifier comities [Sebastiani, 2002] used in text categorisation, the idea is that shortcomings of individual recommenders can be overcome by combining tag prediction scores generated by different approaches. The individual parts of the hybrid can include folksonomy-based as well as content-based approaches. In [Lipczak and Milios, 2010a], Lipczak et al. use keyword extraction techniques to extract a set of un-personalised tags from the document content, and then personalise this set of candidate tags by combining it with the query user's list of preferred tags. The challenge with hybrid approaches is how to combine the recommendations given by the individual components of the system to achieve the best effect. Several methodologies for combining recommenders have been examined in recommender systems [Burke, 2002]. For tag recommendation, a widely-used method [Jäschke et al., 2008; Lipczak and Milios, 2010a; Gemmell et al., 2010] of combining two recommenders it to take a linear combination of their tag prediction scores. The final recommendation set includes all tags which appear in one or both of the sets. Given two sets of prediction scores A and B, generated by different recommenders, the score of each tag t in the final recommendations is a weighted sum of the tag's scores in A and in B. The effect is

that scores of tags which appear in both source sets are increased relative to scores of tags which appear only in A or only in B. This combination is referred to as the linear combination [Lipczak and Milios, 2010a; Gemmell et al., 2010] or the "p-mix" [Jäschke et al., 2008] of prediction sets. The combination weight that determines the balance in importance given to scores from A and B is set by a parameter b where $0 \leq b \leq 1$. The optimal setting for b is usually found by performing tuning runs on an additional tuning split of the data and the best found setting is used when generating predictions for the test set. A thorough examination of the learning process for combination weights in hybrid tag recommenders is carried out in [Lipczak and Milios, 2010a].

2.5 Evaluation Methodologies

The tag recommendation task is to recommend a set of tags for a query post $(u_q, d_q, \emptyset)$. A tag is successfully recommended if user u_q chooses it tagging document d_q, otherwise it is a wrong recommendation. The number of tags that are recommended can be variable but is usually set to a fixed number between 1 and 10 across all recommendations. Tag recommendation algorithms can be evaluated online in a live social bookmarking system or offline by using a holdout set for which tags are to be predicted. Live evaluation is rarely used in tag recommendation research since it requires access to a live social tagging system with many active users and is a lengthy process where additional non-algorithmic influences also play a role. One such live evaluation was carried out on the BibSonomy website in [Jäschke et al., 2009]. It is much more common to evaluate tag recommendation algorithms offline by splitting available social tagging datasets into a training set and a test (holdout) set. Models are built on the training set and recommendations are generated for query posts from the test set. To evaluate the success rate of recommendations, the predicted tag sets are compared to the real tag sets that were chosen by users for query posts. The accuracy of algorithms is then measured based on how many of the correct tags were recommended.

2.5.1 Training and Test Split

Two methodologies are mainly used to construct the training and test data for evaluating recommenders. The split is either done by date as this is closest to the reality of the application, or by using a leave-one-out approach. In the date-split, the set of posts is split into a training and a test set on a predefined cut-off date, putting all posts entered before the chosen date into the training set and all posts

entered after the chosen date into the test set. This approach models the real-world scenario of tag recommendation as closely as possible and was used to compare tag recommender submissions to the ECML PKDD Discovery Challenge of 2009 [Eisterlehner et al., 2009]. An alternative approach is to construct a leave-one-out split per user, where for each user one post is left out from the training data and put in the test data. This approach can be used in multi-fold cross-validation [Mitchell, 1997] where several training and test sets are created, with each test fold containing a random test post for each user. The final results are then given as the average accuracy over folds. However, the leave-one-out approach constructs an easier tag recommendation problem than the more realistic date split, and is only applicable where all users have multiple posts in the dataset, ie. datasets at post-core level 2 or greater (post-core processing is described in Section 2.6.2).

2.5.2 Success Measures

The standard measures for evaluating the quality of tag recommendations are precision, recall and F1. Since the tag recommendation problem is to recommend a set of tags where the real number of tags can vary in each post, a predefined number of tags to be recommended is usually set and the success rates are given at that size, for example recall@5 is the recall achieved when recommending five tags. Precision measures the ratio between the number of correct predictions and the total number of predictions made. It is given by

$$precision = \frac{TP}{(TP + FP)}$$

where TP (true positives) is the number of correctly recommended tags and FP (false positives) is the number of wrong recommendations. Recall measures the ratio between the number of correct predictions and the actual number of tags assigned by the user.

$$recall = \frac{TP}{(TP + FN)}$$

where FN are false negatives, the number of correct tags which were not recommended. In order to combine the precision and recall values either the precision/recall break-even point or the F1 measure can be used. The precision/recall break-even point is defined as the point (in the case of tag recommendation the number of recommended tags) at which precision and recall are equal or closest. It gives a single number as the success measure. The F1 measure is a combination of precision and recall that gives equal importance to both of them. F1 can be

calculated at all numbers of recommended tags and is defined by

$$\text{F1} = \frac{2 * precision * recall}{precision + recall}$$

Precision, recall and F1 are usually first calculated per query post, and then averaged over all posts to give the overall success rate (macro-averaging).

Given the characteristics of social tagging systems, recall is much more important to the user than precision. While recall penalises false negatives, so considers the number of tags that the user wanted but were not recommended, precision penalises false positives, by giving importance to how many wrong guesses were made. As long as the number of recommended tags displayed per post is sensibly small ($\leq$10), the cost of false positives is minimal. The user will read the wrongly recommended tag, ignore it and move on to choose other tags. The cost of making another guess is thus very small, and the exact number of guesses is not as important as long as some correct recommendations are made within a reasonable amount of guesses ($\leq$10). The cost of a false negative, an omitted correct tag, is a lot higher since the user has to think of and type in the desired tag himself, which means spending time and effort. We thus want to have recommenders which find as many correct tags as possible within a reasonable number of guesses, where the exact number of guesses is not as important as having a large coverage (or recall) of the true tag set. Recall@N, where N is usually set to 5 is widely-used and believed to be one of the most meaningful measures to compare tag recommenders. However, with recall@N one has to be aware that for posts where the number of true tags is greater than N, recall@N can never reach 1 and will report a sub-optimal success rate even when all N recommended tags for a post are correct, and the precision@N is equal to 1.

2.6 Social Bookmarking Websites and Dataset Retrieval

The effectiveness of tag recommendation algorithms is evaluated on datasets obtained from social bookmarking websites, where the most prominent ones for tagging textual documents are CiteULike, BibSonomy and Delicious. CiteULike is a publication bookmarking system for sharing and collaboratively tagging research papers. CiteULike provides official snapshots of their social tagging dataset including all tag assignments in the system to date with timestamps for each tag assignment. Additionally to the social tagging data, the CiteULike website stores the BibTex entries of the research papers that are tagged in the system. The BibTeX entries corresponding to documents in the snapshots are not available to download as a package, but

can be identified through the *citeulike-article-id* field in the snapshots and downloaded from the website. BibSonomy [Benz et al., 2010] is a social bookmarking system for both websites and publications. The system is split into two separate parts: BibSonomy Bookmark and Bibsonomy BibTeX, corresponding to website bookmarks and publication bookmarks respectively. Official snapshots of the BibSonomy dataset are available separately for the Bookmark and BibTeX part of the website. The BibSonomy snapshots additionally include information in meta-data fields such as *title* and *description* for websites and *title*, *author* and *bibtexAbstract* for publications. Delicious is by far the biggest existing social bookmarking system and allows users to tag websites. While CiteULike and BibSonomy provide official snapshots of their complete datasets, Delicious does not. The Delicious datasets used in research are obtained by partially crawling the delicious website, since a full crawl is not feasible due to the large size and technical restrictions. We give the tagging data sizes and statistics of CiteULike and BibSonomy snapshots in Tables 2.1 and 2.2 respectively, and address the crawling of Delicious in the next section. In our experimental evaluation in the next chapters we use the CiteULike 2012-05 snapshot and the Delicious crawl of [Hotho et al., 2006a] that was kindly provided to us by the authors. For BibSonomy we use a cleaned version of the full 2012-07 snapshot given in Table 2.2, where spam and imported posts have been removed by the authors and administrators of the BibSonomy system.

CiteULike 2012	
Posts	4,863,375
Tag Assignments	17,145,727
Users	115,782
Documents	3,731,020
Tags	752,295

Table 2.1: Citeulike 2012-05 Snapshot

Bibsonomy 2012	Bookmark	BibTeX	Total
Posts	359,641	109,984	469,625
Tag Assignments	1,344,760	395,699	1,740,459
Users	4,996	4,777	9,773
Documents	311,978	94,427	406,405
Tags	89,596	57,526	147,122

Table 2.2: BibSonomy 2012-07 Snapshot

2.6.1 Crawls

The crawling methods used to obtain Delicious datasets used in research are effectively implementations of sampling techniques. Several different Delicious samples, which we list in Table 2.3, have been used in tag recommendation literature. The complete size of the actual Delicious dataset is not known. The last official numbers provided by Delicious are from September 2008 where it was claimed to have 180 million unique documents. All crawling methods used to obtain the existing delicious datasets can be described as user-based sampling. The general method used is to first sample a list of users and then download the complete (or near-complete) user profiles for the sampled users by retrieving all of their posts, including the user, document and tags. The difference in the crawling methods lies in how the list of users is sampled. In most of the crawls, the initial sample of users is obtained by a breadth-first [Kurant et al., 2011] exploration of the folksonomy graph where only user nodes are added to the sample instead of adding all encountered nodes. [Hotho et al., 2006a] start with the most popular document of delicious to obtain the list of users and tags associated with this document (which resulted in about 6900 users and 700 tags). They then recursively explore users and documents related to these items and monitor the start page of delicious for further users and resources. The result is a list of 75,242 user-names which is taken as the sample list of users, and for each of these users the tagging data related to their first 10,000 posts is downloaded to give the dataset. [Shepitsen et al., 2008] iterate over users who have tagged the most popular documents and recursively expand the list of users via shared documents to create the sample list of users. The complete user profiles (all posts) of 29,918 users found in this manner are then downloaded to give the dataset. For their experiments, Shepitsen et al apply a further user-based sampling to the dataset, creating two samples of 5,000 users each on which the experiments are conducted. A similar sampling approach is used by [Gemmell et al., 2009], where the sample list of users is created by starting from users who have used the most popular tags and exploring further users over the "Network" and "Fan" relationships which exist in Delicious. After downloading the complete user profiles of the resulting 524,790 users, the dataset used in the experiments is created by taking another user-based sample which consists of 10% of these users taken at random. (Statistics of this dataset are not listed in Table 2.3 since Gemmell et al also apply post-core processing and finally use a post-core at level 20 in their experiments.) [Wetzker et al., 2008] use all posts containing the tag "web2.0" as a starting point and recursively explore all related tags. From the resulting dataset, the sample list of users is created by selecting the users with the most posts. The

complete user profiles of these users are downloaded to give a large crawl. The large crawl has not been used for a tag recommendation task in [Wetzker et al., 2008] but rather to conduct an extensive statistical analysis of Delicious. For application to the tag recommendation task Wetzker et al. apply further sampling to the crawl in [Wetzker et al., 2010]. The sample is created by first removing posts they believe to be spam and then limiting the time range of the dataset to the first 16 months (11.2003-12.2004). [Lipczak and Milios, 2011] create and use a dataset which is the overlap of the crawls from [Wetzker et al., 2008] and [Bender et al., 2008]. This is done because one of the crawls does not contain document titles while the other does not contain the time-stamps of tag assignments and Lipczak et al. require both.

	Posts	Tag Assignments
Bender et al. [Bender et al., 2008]	4,582,773	not given
Hotho et al. [Hotho et al., 2006a]	7,698,653	17,780,260
Lipczak et al. [Lipczak and Milios, 2011]	$\approx$ 9,000,000	not given
Shepitsen et al. [Shepitsen et al., 2008]	not given	47,184,492
Wetzker et al. [Wetzker et al., 2010]	1,909,687	3,906,207

Table 2.3: Delicious Crawls

Sampling Bias The sampling methods used to crawl Delicious have a sample bias towards popular entries, as either a popular node or a set of popular nodes is taken as the starting point for graph exploration. Additionally, the method applied to sample the initial list of users is usually breadth-first sampling which is known to be biased towards highly connected nodes [Kurant et al., 2011]. More advanced sampling techniques such as Metropolis Hastings Random Walk (MHRW) [Gjoka et al., 2011], MHRW-DA [Lee et al., 2012] or Albatross Sampling [Jin et al., 2011] have not yet been applied to crawl tag recommendation datasets. Due to the bias in the sampling of the Delicious datasets, the test set used to evaluate the accuracy of tag recommendation algorithms will also be biased towards popular tags. When using a test set that has this sample bias, situations might arise where tag recommendation algorithms weighted towards recommending popular tags produce a higher prediction accuracy than if they were evaluated on an un-biased test set. Evidence of this can be seen in our evaluation in Chapter 3 where we conduct experiments using the Delicious crawl of [Hotho et al., 2006a]. We show that a baseline recommender suggesting the most popular in the system for every test post produces relatively high results on the Delicious dataset while having a low accuracy for CiteULike and BibSonomy where the test sets are from complete snapshots without a sample bias.

While the focus of this work is on the tag recommendation algorithms themselves we plan to further explore the sampling methods and sampling bias of social tagging datasets in the future.

2.6.2 Post-Core Processing

Folksonomy-based tag recommendation algorithms are often evaluated on post-core subsets of the full social tagging data. In a post-core at level k, only posts are included where the user, document and all tags appear in at least k posts of the dataset. Post-core processing is used to provide a denser datasets for models to be built on and also minimise the chance that new items users, documents or tags will be encountered in the test set. Since folksonomy-based recommenders cannot successfully recommend tags for new documents or users, post-core processing is used as a means to evaluate the algorithms on test cases where sufficient information is available. However, post-cores only make up a small subset of the full folksonomy in which the majority of posts is not included. While the new user problem is not as big an issue as each user is only new when submitting their first post, the new document problem is very prominent in social tagging datasets. The majority of documents in unpruned datasets is only tagged by one user [Wetzker et al., 2008; Lipczak and Milios, 2010b], and thus many query documents in the full test set have no previous tagging information associated with them. Since content-based approaches can address the new document problem and recommend tags for previously untagged documents based on their content, they are usually evaluated on the unpruned datasets, encompassing the full real-world tag recommendation problem.

2.7 Conclusion

Existing tag recommendation approaches can be categorised by the source and scope of information that they consider. In folksonomy-based approaches the information source is the social tagging data, while in content-based approaches the focus is on analysing the textual content of documents. Folksonomy-based approaches achieve high accuracy when recommending tags for query documents (and query users) that already have some tagging information associated with them in the historical data. However, they are not successful at recommending tags for previously untagged documents. Content-based approaches can address this new document problem by representing documents based on their textual content instead of their tagging information in the folksonomy data. Approaches analysing document content originate from the field of text analysis and traditionally address an un-personalised docu-

ment labelling task. In order to recommend tags related to the content of the query document as well as personalised to the preferences of the query user, content-based and folksonomy-based approaches are combined as hybrids. The hybridisation is usually done by first generating predictions from the individual recommenders and then blending their recommendation sets. The contributions of our work regarding document content, presented in the next chapter, approach this hybridisation problem from a different perspective by including content into folksonomy-based approaches at the algorithmic level, before predictions are generated. This results in novel content-aware, folksonomy-based recommenders. Our approach of including content into graph ranking methods produces recommenders that have the ability to analyse the full scope of tagging information contained in the folksonomy as well as considering content in the tag recommendation process.

Most tag recommendation approaches are adaptations of algorithms that originate from research in traditional recommender systems and information retrieval. The original algorithms were designed to address problems where the datasets have a lower dimensionality of types of items as well as types of relations between items. The algorithms were adapted to the tag recommendation task in order to enable them to model the 3-dimensional folksonomy data, and in order to address the problem of recommending tags related to two query items instead of one, the query user and query document. Graph-based ranking approaches have received a lot of attention in the tag recommendation community due to their ability to model and analyse the full scope of available information in the social tagging data. Much research has been done on adapting the algorithms to efficiently handle the three dimensions of the folksonomy. However, a thorough examination of the underlying assumptions that made graph-based ranking successful in information retrieval and whether these assumptions directly apply to folksonomies has not yet been carried out. In Chapter 4 we conduct a detailed analysis of the FolkRank graph ranking algorithm and propose novel adaptations to the graph model as well as weight spreading approach used in FolkRank. We examine implicit assumptions made by the tri-partite graph model of the folksonomy and show how these assumptions influence the tag recommendation process. Moreover, we analyse the iterative weight spreading approach of FolkRank and highlight issues that arise due to the specific structure of the folksonomy graph. As part of our contributions we propose and alternative weight spreading algorithm that addresses the issues we have identified and also has a lower computational complexity than FolkRank's iterative approach. The evaluation of our hypotheses reveals novel insights and limitations concerning the applicability and advantages of graph-based ranking methods for

tag recommendation. In Chapter 5 we expand on these insights and analyse the existence of negative feedback in the folksonomy graph structure. Implicit negative feedback has been explored in recommender systems where all non-co-occurrences in 2-dimensional data are assumed to imply negative relationships. By taking the 3-dimensional relationships in the folksonomy into consideration, we show that a stronger indication of negative feedback can be extracted from social tagging data than in 2-dimensional datasets. We evaluate the existence of negative feedback in folksonomies, and the reliability of our approach for extracting implicit negative feedback from the graph structure. We show that the existence of negative feedback is a factor that can hinder the ability of graph-based approaches to make more accurate tag predictions, and propose that negative feedback has to be taken into account when developing graph-based tag recommendation approaches in the future. Overall our contributions include several content-aware, folksonomy-based recommenders, a thorough examination of graph modelling and ranking methods for folksonomies and new insights into the fundamental characteristics of social tagging datasets.

Chapter 3

Content-Awareness

In this chapter we present our content-aware extensions of folksonomy-based tag recommendation algorithms. We propose and compare two methods of including content. Our first approach is to include content at the word level directly into the folksonomy model. We represent documents by their content words and model relationships between users and words, words and tags, and users and tags. A query post is then treated as consisting not of a query user and query document, but instead as consisting of the query user and a collection of query words. Our second alternative method includes content indirectly at the document level. In this approach the underlying model of the folksonomy-based recommenders is not adapted with content data. However, content is included when constructing the query passed to the algorithms. By using content-based similarity, a neighbourhood of already tagged training documents most similar to the query document is constructed. The documents from the content-based neighbourhood are then included as part of the query passed to the folksonomy-based recommenders. We apply our two alternative methodologies for including content to the graph-based FolkRank algorithm and also to a simpler co-occurrence recommender.

Our extensions make the folksonomy-based recommenders applicable to unpruned, real-world tagging data and address the new document problem in tag recommendation. Including content gives a significant increase in the recommendation accuracy of FolkRank as well as our simpler co-occurrence recommender on unpruned tagging datasets.

3.1 The Need for Content

Test sets generated from real-world tagging datasets contain a large proportion of new, previously untagged query documents [Lipczak and Milios, 2010b; Wetzker et al., 2008]. A drawback of purely folksonomy-based recommenders such as FolkRank is that they can only generate successful recommendations for query posts where the user and document are already known to the system. It is required that the query user has already tagged at least one document, and that the query document has been tagged by at least one user in the past. This makes solely folksonomy-based recommenders only applicable to post-core subsets of tagging data of level 2 or higher, where most (if not all) query users and documents are known to the system and already have some tags assigned to them. When trying to apply these methods to a query post with a new user and/or new document they default to recommending either the most popular tags of the user, the most popular tags of the document or the most popular tags across all posts, depending on whether both the user and document are new or not. While the new user problem is not as prominent since each user is only new during his first post, the new document problem is present in the majority of query posts in real-world tagging data. Post-cores of level 2 or higher only capture a fraction of the real-world tag recommendation problem. In order to recommend tags for new documents, approaches are required which model documents not only based on the tags assigned to them in the past (if any), but also their content.

3.2 Folksonomy-Based Recommenders Without Content

To evaluate the impact of including content into the recommendation process, we first analyse the folksonomy-based recommenders without content. These are FolkRank and a simple co-occurrence approach which we call CoOcc. While FolkRank analyses the full folksonomy graph when generating recommendations, the simple CoOcc recommender only considers the immediate neighbourhood of the query and does not attempt to exploit the deeper graph. By using CoOcc as a baseline, we can evaluate whether or not exploring the deeper graph with FolkRank provides a benefit over limiting the analysis to the immediate co-occurrence neighbourhood of the query alone.

3.2.1 FolkRank

FolkRank [Hotho et al., 2006a; Jäschke et al., 2007] is a graph-based ranking algorithm which is modelled based on Google's PageRank. Users, documents and tags are represented as nodes $v \in V$ in an undirected tri-partite graph $G = (V, E)$, where all co-occurrences of users and documents, users and tags, and documents and tags are edges $e \in E$ between the corresponding nodes. The weight of the edge between two nodes depends on the number of their co-occurrences, given as the number of tag assignments that both nodes appear in. For example if a user u has used a tag t for two documents, there would be two tag assignments (u, d_1, t) and (u, d_2, t) in the folksonomy, and in G the weight of the edge between the two nodes representing u and t would be set equal to two.

The importance or rank of each node in FolkRank is calculated by an iterative weight-spreading algorithm, in a similar fashion to PageRank. The weights of all nodes are given in the weight vector $\vec{w}$ which has one entry per node and is computed by the weight spreading function

$$\vec{w} \leftarrow (1 - d)M\vec{w} + d\vec{p}$$

where M is the row-stochastic version of the adjacency matrix of graph G, $\vec{p}$ is the preference vector, and the dampening factor $0 < d \leq 1$ determines the influence of $\vec{p}$. The preference vector $\vec{p}$ is used as a means to personalise the recommendations to the query user and query document, and is set to give these query nodes in the graph a higher preference weight compared to other nodes. The dampening factor d sets the balance between personal preference and global importance when calculating the node weights. After constructing the folksonomy graph, the tag ranking procedure with FolkRank is as follows for each test post.

1. Initialise each node in the graph with a random starting weight so that the total sum of node weights in the graph is equal to a predefined parameter TW (total weight).

2. Set the preference vector giving the query user and document a higher weight than other nodes in the graph, and so that the sum of weights in the preference vector, denoted by PW (preference weight), is equal to the total weight in the graph TW.

3. Perform iterative weight spreading until node weights converge. The end condition is that the sum of absolute change in node weights during one iteration is smaller than a predefined fraction of the total weight TW.

4. Select the nodes which represent tags and rank them by node weight, where the tag node with the highest weight is given the best ranking.

When setting the weights of the preference vector it is important that the sum of preference weights is equal to the total sum of node weights in the graph. This ensures that the total weight in the graph stays constant over weight-spreading iterations; that no factors other than parameter d impact the amount of personalisation; and that the end condition of iterative spreading will work as intended. FolkRank can generate a global non-personalised ranking and a personalised ranking of all nodes in the graph, depending on the values set in the preference vector. For a global ranking, the entries in $\vec{p}$ for all nodes are set to the same value. In order to generate personalised recommendations for a query post $(u_q, d_q, \emptyset)$, $\vec{p}$ is set so that higher preference weights are given to the query user u_q and query document d_q, compared to other nodes in the graph which are set to have a uniformly small (non-zero) preference weight [Hotho et al., 2006a].

In the later version of FolkRank presented in [Jäschke et al., 2007], a differential approach is used to calculate the final scores of the tag nodes. The reason for this given in [Jäschke et al., 2007] is that due to the undirected nature of the graph, weights are distributed in one direction of an edge in one iteration, and then distributed back along the same edge in the next. Jäschke et al. claim this makes it very difficult for nodes other than the ones with high edge degree to become highly ranked, no matter what the preference vector is. In order to generate recommendations for a query post $(u_q, d_q, \emptyset)$, personalised tag ranks are first calculated by setting a preference vector which gives higher weights to u_q and d_q. From the result, another set of tag ranks is subtracted which is generated with a uniform preference vector (giving equal preference to all nodes in the graph) in order to calculate the final tag scores. However, this differential solution is further discussed in [Kim and El Saddik, 2011] and [Ramezani et al., 2010], where it is suggested that other very similar adaptations of PageRank (and FolkRank) do not have this problem and produce comparable results in only one run with a personalised preference vector. Kim et al. show in [Kim and El Saddik, 2011] that equivalent tag rankings and prediction results to the differential approach can be achieved by doing one personalised run where the preference weights of all non-query nodes are set to zero, instead of to a uniformly small positive weight as is the case in the personalised run of the differential approach.

An unexplored question in FolkRank is whether the same amount of preference weight should be given to the query user and query document. In the original FolkRank approach the balance of preference weight between the query user and

document is not specified explicitly via a parameter but instead is determined implicitly by the ratio of user nodes to document nodes in the graph [Jäschke et al., 2007]. In general this leads to the query document receiving more preference weight than the query user since there are usually more documents than users in a folksonomy, however it is determined entirely by the data. We introduce a parameter b into our FolkRank-based approaches which allows us to control how the total preference weight is distributed between the query user and document. The preference weights of the query user u_q and query document d_q are then given by

$$\mathrm{pw}(u_q) = b * PW$$

$$\mathrm{pw}(d_q) = (1 - b) * PW$$

where $0 \leq b \leq 1$, PW is the total preference weight and $PW = TW$ since we set the total preference weight equal to the total weight in the graph. If we set $b = \frac{|U|}{|U|+|D|}$, where U is the set of users and D is the set of documents, then we have an equivalent strategy to that used in the original FolkRank.

3.2.2 Co-Occurrence Approach

The co-occurrence recommender we use as a baseline is a combination of user-related and document-related tags. We combine the sets of tags co-occurring with the query user and the set of tags co-occurring with the query document. For query posts where the set of user and document-related tags does not contain the wanted number of recommendations (or is empty), we add the most popular tags in the data overall to fill the recommendation set to the required size.

User Tags (UT)

The User Tags recommender bases its recommendations on the query user only and ignores the document to be tagged. For a query post $(u_q, d_q, \emptyset)$, the set of candidate tags consists of the past tags of the query user u_q. The score for each of the candidate tags t is the co-occurrence count of u_q and t divided by the total number of posts of u_q, and can be seen as the probability that u_q will use t for any document. We use the notation $\mathrm{UT}(u_q, t)$ to denote the prediction score of tag t for user u_q, calculated as

$$\mathrm{UT}(u_q, t) = \frac{|(u_q, d_\exists, t) \in A|}{|(u_q, d_\exists, S_\exists) \in P|}$$

where $d_\exists$ is any document and $S_\exists$ is any tag set. The numerator of the fraction is the number of co-occurrences of u_q and t in the set of tag assignments A, and the

denominator is the total number of posts made by u_q.

Document Tags (DT)

Analogous to the User Tags recommender, we calculate a tag co-occurrence probability for a document d_q as

$$\mathrm{DT}(d_q, t) = \frac{|(u_\exists, d_q, t) \in A|}{|(u_\exists, d_q, S_\exists) \in P|}$$

where $u_\exists$ is any user and $S_\exists$ is any tag set. The numerator is the number of co-occurrences of d and t, and the denominator is the total number of posts containing d. The tag score $\mathrm{DT}(d_q, t)$ represent the probability that tag t will be assigned to d_q by any user.

Most Popular Tags (MP)

The simplest baseline recommender is the Most Popular Tags recommender which recommends the same set of tags for all test posts, the recommendation consisting of the top N most frequently used tags in the system. The MP score for a tag t is the probability that t will be used in any post by any user for any document, and is calculated as

$$\mathrm{MP}(t) = \frac{|(u_\exists, d_\exists, t) \in A|}{|P|}$$

where the numerator is the number of tag assignments containing tag t, and the denominator is the total number of posts.

Co-Occurrence Recommender (CoOcc)

To recommend tags which are related to the document as well as personalised to the user's preferences, the tag prediction scores of the User Tags (UT) and the Document Tags (DT) recommenders are combined. To combine the prediction scores from UT and DT, we apply the standard approach of taking a linear combination of the two prediction sets [Lipczak and Milios, 2010a; Gemmell et al., 2010]. However, we believe that it is important to highlight that the candidate tag set of the UT recommender does not include all tags that are found by DT, and vice versa. Since the candidate tag set of the combined approach includes all tags that appear in either only one or both of the source sets, we refer to the weighed linear combination method of prediction sets as the union. The score of each tag t in the union UT$\cup$DT is a weighted sum of the tag's scores in UT and in DT. The effect is that scores

of tags which appear in both source sets are increased relative to scores of tags which appear only in UT or only in DT. We use the notation $\mathrm{UT}(u_q)$ to denote the candidate tag set of the UT recommender for user u_q, and $\mathrm{UT}(u_q, t)$ to denote the prediction score of tag t for user u_q. Similarly for DT, $\mathrm{DT}(d_q)$ denotes the candidate tag set for document d_q and $\mathrm{DT}(d_q, t)$ denotes the prediction score of tag t. We calculate the score of each tag in the union UT∪DT as

$$\mathrm{UT{\cup}DT}(u_q, d_q, t) = \begin{cases} b * \mathrm{UT}(u_q, t) + (1-b) * \mathrm{DT}(d_q, t) & \text{if } t \in \mathrm{UT}(u_q) \wedge t \in \mathrm{DT}(d_q) \\ b * \mathrm{UT}(u_q, t) & \text{if } t \in \mathrm{UT}(u_q) \wedge t \notin \mathrm{DT}(d_q) \\ (1-b) * \mathrm{DT}(d_q, t) & \text{if } t \notin \mathrm{UT}(u_q) \wedge t \in \mathrm{DT}(d_q) \end{cases}$$

where $0 \leq b \leq 1$ is a parameter that determines the balance in importance given to scores from UT and DT. Before combining the recommendation sets, we normalise the tag scores in each of the two source sets so that they sum to one. To find the optimal setting for parameter b, we tune b on the evaluation set and then use the best value on the test set. Since a query post can contain a user and document that are both new or have only very few tag co-occurrences in the historical data, the UT∪DT recommendation set might not contain a sufficient number of tags in some cases. For these cases, we use the most popular tags (MP) to fill the recommendation set to the required size and produce the final recommendations of the CoOcc recommender. From MP, the tags which are not already included in UT∪DT are appended to the end of the tag rankings of UT∪DT in order of their overall popularity. Tags which are added from MP can thus never outrank existing tags in UT∪DT, and the scores from MP do not influence the scores of existing tags in UT∪DT. The most popular tags are used only as a means to fill the recommendation set to the required size.

A direct comparison of the accuracy achieved with FolkRank and a recommender based on co-occurrence is presented in [Jäschke et al., 2008], where FolkRank is reported to produce significantly better results. Similarly to CoOcc, the co-occurrence recommender used in [Jäschke et al., 2008], called "most popular tags mix", is a weighted combination of tags co-occurring with the query user and tags co-occurring with the query document. However, there are major differences between "most popular tags mix" and our CoOcc approach in how the co-occurrence scores are calculated and how the weights are normalised before combination. The recommender used in [Jäschke et al., 2008] only considers the absolute user-tag and document-tag co-occurrence counts without considering the total number of posts of the user or document, which would correspond to using only the numerator in our UT and DT score calculations. Before combining the user-tag and document-tag

co-occurrences to give the final recommendation set, Jäschke et al. normalise the co-occurrence counts from the two sources to be in the interval $[0, 1]$ so that the highest scoring tag has a score of 1 and the lowest scoring tag a score of 0. However, the distribution of co-occurrence counts across all tags of a user and all tags of a document are not considered. Only the minimum and maximum scores in each of the source sets are taken into account in the calculation of the normalised scores.

3.3 Extension with Content

Here we present our methods for extending the folksonomy-based recommenders with content data. These content-aware recommenders include the textual content of documents in the recommendation process as well as utilising the relationships given in folksonomy data. We can thus overcome the new document problem and make the folksonomy-based recommenders applicable to full real-world datasets. For test posts where the query user is new (as well), we have to default to the most popular tags found to be related to the content of the query document and cannot personalise these to the user, which is acceptable since the user does not have a tagging profile yet. In the following sections we first describe the document content model used and then present our content-aware FolkRank and co-occurrence recommenders.

3.3.1 Document Model

For including data from the content of documents in the tag recommendation algorithms, we consider two sources of content words: document title and full-text content. We convert all words to lower-case, remove stop-words as well as all words which have a length of less than 3 or more than 20 characters, and use the remaining words without stemming. Each document is then represented by a bag-of-words vector of content words with Tf-Idf scores for each word. Tf-Idf stands for *Term frequency - Inverse document frequency* and we compute it as

$$\text{Tf-Idf}(w, d) = \frac{\text{tc}(w, d)}{\sum_{w_l \in d} \text{tc}(w_l, d)} * \log_2 \frac{|D|}{\text{dc}(w, D)}$$

where D is the set of all documents, $\text{tc}(w, d)$ is the term count equal the number of occurrences of word w in document d, and $\text{dc}(w_l, D)$ is the document count equal to the number of documents in the collection containing word w. The left side of the multiplication is the term frequency measuring the fraction of occurrences of word w over occurrences of all words in d. The right side is the $\log_2$ of the inverse

document frequency. The document frequency measures the fraction of documents in collection D that contain word w. We normalise the Tf-Idf scores to sum to 1 per document, as this is required for our content-aware approaches to work correctly.

A factor to consider is that the content data of websites can change over time. The title, content and meta-data of a website which is bookmarked can be updated and differ from one post to the next. This presents a problem, as well as additional data for analysis. The full-text content of the bookmarked document (or website) is only available in the current version at the time of retrieval, however, the BibSonomy dataset provides different versions of the title of a document at the time of each post. Where available, we concatenate the title variations of a document from all its posts and treat the resulting text string as the single document title. This makes the term count measure $\text{tc}(w_l, d_j)$ in our Tf-Idf calculation more powerful as words which persist over several title variations will end up with a higher score than words which only appear in a few of the variations.

3.3.2 Content-Aware FolkRank

Our two alternative methods of including content into FolkRank are to include content words directly into the folksonomy graph model (ContentFolkRank), or to include content information indirectly by using a content-based document similarity measure (SimFolkRank).

ContentFolkRank

ContentFolkRank includes content data directly into the folksonomy graph model. We adapt the folksonomy model to use triplets (*user*, *word*, *tag*) instead of (*user*, *document*, *tag*). Each tag assignment in the training data (u, d, t) is converted to a set of tag assignments with words instead of documents $\{(u, w_1, t), (u, w_2, t), ..., (u, w_x, t)\}$ where each of the words $w \in d$. The preference vector for a query post $(u_q, d_q, \emptyset)$ is then given by $(u_q, w_1, w_2, ..., w_x)$ where each query word $w_q \in d_q$.

Replacing each document node in the graph with a variable number of word nodes presents an issue with the edge weighting scheme of the graph. In the regular FolkRank graph with user, document and tag nodes (Figure 3.1), the weight of the edge between a user u_1 and a tag t_1 depends on the number of documents the user u_1 has tagged with t_1 (in this case 1). If we replace the document node d_1 by multiple word nodes $w_1, w_2, .., w_x$ (Figure 3.2), the weight of the edge between u_1 and t_1 is influenced by the number of words in the document if using the original weighting scheme of FolkRank. In our example the weight of this edge would now be equal to

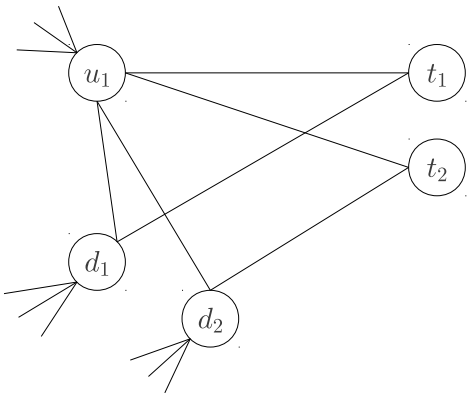


Figure 3.1: FolkRank

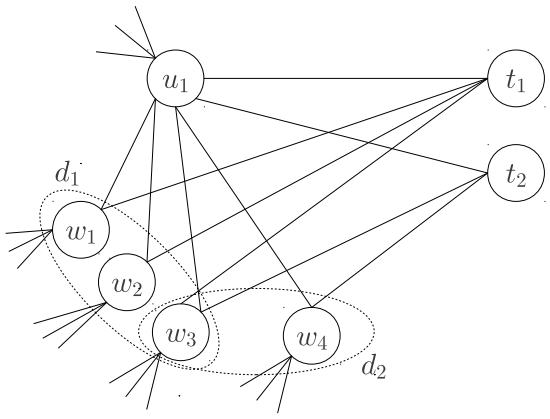


Figure 3.2: ContentFolkRank

three. The problem with this becomes apparent when user u_1 adds a new document to his collection and weight spreading is applied to recommend tags to u_1. If using the regular FolkRank graph, the activation spread from u_1 to each of the tags t_1 and t_2 is equal which makes sense since the user has used each of the tags once. Each of the tags gets an activation weight of $\text{outActivation}(u_1) * \frac{1}{(|\text{edges of } u_1|)}$. If we use the ContentFolkRank graph however, the activation spread from u_1 to t_1 would be $\text{outActivation}(u_1) * \frac{3}{(|\text{edges of } u_1|)}$, while the activation spread to t_2 would only be $\text{outActivation}(u_1)\frac{2}{(|\text{edges of } u_1|)}$. Tag t_1 would thus receive a higher weight than t_2 because d_1 happens to contain more words than d_2.

In order to overcome this problem, we create custom rules for setting the weights of edges connecting different types of nodes, namely user-word edges, word-

tag edges and user-tag edges. In our example, we want the sum of the weights of edges connecting u_1 to any of the words from d_1 to be equal to the sum of the weights of edges connecting u_1 to any of the words from d_2. More generally, we want the sum of the weights of edges connecting any user u to words nodes from any one document d to be equal to a pre-defined constant. This would mean that regardless of the number of words that a document is represented by, the sum of weights of edges connecting u to the word nodes representing d will always be the same. To achieve this and additionally include the varying importance of different content words to the document, we use the Tf-Idf scores of the words in the document. The Tf-Idf scores are normalised to sum up to 1 per document to provide suitable weights for the edges between a user and each of the word nodes of the document. Since several documents, for example d_a and d_b, tagged by the same user u_a can contain the same word w_a, the weight of the edge between u_a and w_a is set to the sum of the normalised Tf-Idf scores of w_a in d_a and d_b. The same holds for edges between word and tag nodes. The weight of the edges between user and tag nodes, u_a and t_a, is set to the number of posts (documents) in which u_a has used t_a. The formulae for calculating the weights of the different types of edges are the following.

User - Word Edges

$$\text{edgeWeight}(u, w) = \sum_{d_j \in \text{Posts}(u,w)} \text{Tf-Idf}(w, d_j)$$

where $\text{Posts}(u, w)$ is the set of posts by user u where the document contained word w.

Word - Tag Edges

$$\text{edgeWeight}(w, t) = \sum_{d_j \in \text{Posts}(w,t)} \text{Tf-Idf}(w, d_j)$$

where $\text{Posts}(w, t)$ is the set of posts tagged with t (by any user) where the document contained word w.

User - Tag Edges

$$\text{edgeWeight}(u, t) = \sum_{\text{Posts}(u,t)} 1$$

where $\text{Posts}(u, t)$ is the set of posts by user u tagged with t.

The preference vector for each test post is given by $(u_q,\ w_1,\ w_2,\ ...,\ w_x)$ where each query word $w \in d_q$. The preference weight of the query user u_q is the

same as in plain FolkRank $\text{pw}(u_q) = b * PW$, while the preference weight for each query word $w \in d_q$ is set proportional to its Tf-Idf score in d_q, and is given by $\text{pw}(w) = (1 - b) * PW * \text{TF-Idf}(w, d_q)$. Since the Tf-Idf weights are normalised to sum 1 per document, the sum of the preference weights of all words $w \in d_q$ is equal to the preference weight that would be attributed to the document in the plain FolkRank approach without content.

SimFolkRank

Our second approach of including content into the recommendation process is to utilise a content-based document similarity measure and include content information implicitly rather than introducing words directly into the graph. The graph model of SimFolkRank does not contain content data itself and documents are represented by document nodes using the original folksonomy graph. However, for each test post, we construct the preference vector to include not only the query document (if it already exists in the graph) but also a predefined number of training documents most similar in content to the query document. The similarity between documents is calculated by cosine similarity of their word vectors with Tf-Ids scores. We construct the neighbourhood of similar documents $N(d_q)$ by selecting the top k training documents with highest cosine similarity to the query document. Documents with similarity equal to zero are not included in $N(d_q)$ even if the size of the resulting neighbourhood is less than k. Once the k-neighbourhood is constructed, we normalise all similarity scores for the query document d_q to sum to one. This ensures that the number of similar training documents with cosine similarity greater than zero does not affect the recommendation process at higher values of k. The final similarity score between the query document d_q and each training document in the neighbourhood $d_s \in N(d_q)$ is given by

$$\text{sim}(d_q, d_s) = \frac{\text{CosSim}(\vec{d_q}, \vec{d_s})}{\sum_{d_j \in N(d_q)} \text{CosSim}(\vec{d_q}, \vec{d_j})}$$

where $\vec{d_q}$, $\vec{d_s}$ and $\vec{d_j}$ are the corresponding word score vectors of documents d_q, d and d_j. In our experiments we evaluate the impact of setting different values for k, the size of the document neighbourhood.

To construct the preference vector for SimFolkRank we include the query user u_q and all documents in the k-neighbourhood of d_q. The preference weight given to each document from the neighbourhood $d_s \in N(d_q)$ is a function of its similarity to the query document d_q and is given by

$$\text{pw}(d_s) = \text{sim}(d_s, d_q) * (1 - b) * PW$$

where $\text{sim}(d_s, d_q)$ is the content-based similarity between d_s and d_q normalised to sum to one, parameter b determines the balance in preference weight between the query user and document, and PW is the total preference weight. The preference weight of the query user is the same as before: $\text{pw}(u_q) = b * PW$.

3.3.3 Content-Aware Co-Occurence

We present our extensions of co-occurrence approaches with content data. Analogous to our extension of FolkRank, our two content-aware approaches are to include content words directly into the co-occurrence model (ContentCoOcc), or to include content information indirectly via document similarity (SimCoOcc). Similarly to the plain CoOcc recommender, the approaches are combinations of user-related and document-related tags. The user-related recommendations do not incorporate content data and are given by User Tags (UT). The document-related recommendations are constructed by content-based alternatives to DT.

ContentCoOcc

The document side of the plain CoOcc recommender, Document Tags (DT), can only recommend tags for documents which have already been tagged in the historical data. To be able to recommend tags for previously untagged documents, we propose a content-based co-occurrence method, Word Tags, that can be used instead of DT. The Word Tags (WT) approach is based on word-tag co-occurrence rather than document-tag co-occurrence. From the binary co-occurrence matrix of documents and tags $D \times T$, we construct a co-occurrence matrix of document content words and tags $W \times T$, where each entry is a decimal number representing the strength of the relationship between word w and tag t, calculated by

$$\text{weight}(w, t) = \sum_{d_j \in \text{Posts}(w,t)} \text{Tf-Idf}(w, d_j)$$

where $\text{Posts}(w, t)$ is the set of posts tagged with t where the document d_j contained word w. To recommend tags for a query document d_q, the set of candidate tags consists of all tags related to at least one of the words in d_q with a weight of greater than zero. The score for each tag in the Word Tag (WT) recommender is influenced by the Tf-Idf scores of the words in the query document and weight of the relationship between these words and the tag. It is calculated by:

$$\text{WT}(d_q, t) = \sum_{w_l \in d_q} (\text{Tf-Idf}(w_l, d_q) * \text{weight}(w_l, t))$$

In order to personalise the content-based recommendations to the user, we combine the recommendation set generated by the Word Tags approach with User Tags. Analogous to UT∪DT in Section 3.2.2, we take the union UT∪WT where the score of each tag t is given by

$$\mathrm{UT}\cup\mathrm{WT}(u_q, d_q, t) = \begin{cases} b * \mathrm{UT}(u_q, t) + (1-b) * \mathrm{WT}(d_q, t) & \text{if } t \in \mathrm{UT}(u_q) \wedge t \in \mathrm{WT}(d_q) \\ b * \mathrm{UT}(u_q, t) & \text{if } t \in \mathrm{UT}(u_q) \wedge t \notin \mathrm{WT}(d_q) \\ (1-b) * \mathrm{WT}(d_q, t) & \text{if } t \notin \mathrm{UT}(u_q) \wedge t \in \mathrm{WT}(d_q) \end{cases}$$

The final recommendation set of the ContentCoOcc recommender is that of UT∪WT, filled with most popular tags (MP) for query posts where UT∪WT does not contain the wanted number of tag recommmendations.

SimCoOcc

Our second content-aware co-occurrence approach includes tags co-occurring with the query user (User Tags) and tags co-occurring with documents that are similar in content to the query document. To construct the document-related recommendation set we propose the Similar Document Tags (SDT) approach which utilises the document-tag co-occurrence as well as the content-based similarity between documents. Given a query document d_q, we first construct the neighbourhood $N(d_q)$ of training documents similar in content to d_q. The score for each candidate tag t related to documents in $N(d_q)$ is then calculated by

$$\mathrm{SDT}(d_q, t) = \sum_{d_j \in N(d_q)} \mathrm{sim}(d_q, d_j) * \mathrm{DT}(d_j, t)$$

where $\mathrm{sim}(d_q, d_j)$ is the similarity between d_q and d_j, and $\mathrm{DT}(d_j, t)$ is the co-occurrence score of document d_j with tag t, given by the DT recommender. As before, we personalise the content-based recommendation set to the user by taking the union UT∪SDT, and then append tags from MP if needed to produce the final recommendations of SimCoOcc.

3.4 Experimental Setup

3.4.1 Datasets

Our datasets consist of tagging data from the social bookmarking websites CiteULike, Delicious and BibSonomy, and additionally downloaded content data for our

content-aware recommenders. Official snapshots of CiteULike and BibSonomy are available on their respective websites. For CiteULike we use the 2012-05-01 snapshot, and for BibSonomy we use a cleaned version of the 2012-07-01 snapshot that is available upon request, where spam and imported posts were removed by the authors of BibSonomy [Benz et al., 2010]. The BibSonomy social bookmarking website and dataset is split into two separate sections: BibSonomy Bookmark which are website bookmarks and BibSonomy BibTeX which are publication bookmarks. We treat these two subsets of BibSonomy as separate datasets in our experiments. Delicious does not provide snapshots of their data. Here we use a dataset that was obtained in [Hotho et al., 2006a] through crawling the Delicious website in 2005, as described in Section 2.6.1.

Additionally, we downloaded all of the available pages from the URLs in the Delcious and BibSonomy Bookmark datasets, and all of the BibTeX entries for CiteULike. For our content-aware recommenders the two content data sources we evaluate are the title and fulltext for websites, and the title and abstract for publications. The Delicious crawl, and the BibSonomy Bookmark and BibSonomy BibTeX datasets provide the titles of documents. For CiteULike we extracted the titles from the downloaded BibTeX entries. The fulltext content for Delicious and BibSonomy Bookmark is the page text of the bookmarked websites, which we extracted from the downloaded pages. For CiteULike and BibSonomy BibTeX, where the bookmarked documents are publications, we use the abstracts from the BibTeX entries as the fulltext content.

3.4.2 Pre-Processing

We pre-processed all of the datasets by casting all tags to lower case, removing duplicate tag assignments that might occur as a result of this, and removing posts which have no tags. Additionally, for CiteULike there are some automatically generated tags which occur very frequently. In order to clean the dataset of these tags, we removed all tag assignments where the tag equals “no-tag” or “bibtex-import”, or matches the regular expressions “*file-import*” or “*import-*”.

3.4.3 Evaluation Metrics

We use recall@N, precision@N and F1@N as our success measures, where N is the predefined number of tags to be recommended. Recall measures the ratio of correct recommendations to the number of true tags of a test post, whereas precision measures the ratio of correct to false recommendations made. Recall and precision

are given by

$$recall = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$precision = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

where TP (true positives) is the number of correct tags recommended, FP (false positives) is the number of wrong recommendations and FN (false negatives) is the number of true tags which were not recommended. F1 is a combination of recall and precision and is given by

$$\text{F1} = \frac{2 * precision * recall}{precision + recall}$$

Since we believe recall to be more important than precision in the context of tag recommendation, as long as N is kept reasonably low (<=10), we use recall in the evaluation phase to identify the best recommenders and configurations. We then give recall as well as F1 for the final results.

3.4.4 Training and Test Sets for Unpruned Tagging Data

To construct training and test sets for the experiments on full unpruned tagging data, we use the following date-split approach for each of the datasets. The test set consist of all posts in the most recent two months of the data which provides us with a large enough test set size. The resulting numbers of test posts are 76,491 for CiteULike, 1.7M for Delicious, 9,506 for BibSonomy Bookmark and 2,843 for BibSonomy BibTeX. The training set is a sample of the data prior to the two test months. We use a sample and not the complete historical data for our training set since the FolkRank-type algorithms have a high computational complexity and expense. Note that we only apply sampling for the training dataset, while in the test set all posts made in the test time-frame are included.

The aim of our sampling methodology for the training set is to achieve a small enough sample size for our models to generate recommendations within a reasonable time while introducing as little bias into the models as possible. To create the training sample we start by selecting the 12 months of data prior to the test months. Social tagging datasets have been shown to be time-sensitive where popular post topics as well as users' interests change over time, and we believe that posts which are older than a year from the test period provide less predictive data for generating recommendations. We then take a stratified sample of documents, where the stratification is based on the number of posts that the documents appears in.

Finally, we retrieve all posts related to the sampled documents to create our training posts sample. This approach ensures that our training sample contains documents which are tagged frequently as well as documents which are tagged infrequently, and reduces the bias towards documents which are only tagged once that would exist if sampling the documents uniformly at random. The resulting sample has the same distribution of documents over number of posts as the full dataset. We employ this approach of first sampling documents and then retrieving the related posts since the number of documents and the resulting size of the content data is the limiting factor which impacts recommendation speed the most in our content-based approaches. Moreover, documents don't suffer from other issues that exist when sampling users or tags and then retrieving all related posts. With users, the number of posts per user varies much more than the number of posts per document, partly due to some users using bulk imports and automatic post submission plug-ins which make them much more frequent users of the system than others. With tags, there is also much more variance in the number of posts per tag than with documents, where the issue is that tags such as "toread", which hold no collaborative value have, a high number of related posts.

We aim to find a sample size which strikes a good balance between improving the recommendation speed of the algorithm and reducing sample bias. We want to select a sample size at which we achieve a low variation in prediction quality for different samples of the same size, and at which the increase to a larger sample is not justified by a significant improvement in prediction quality. To find an appropriate sample size, we create 5 different training samples of the same size and evaluate models built on them against the test set to find the amount of variation in prediction results. We then increase the sample size to a larger value and repeat the same process, until we are confident that the sample size gives a low variation in different samples of the same size and the move to a larger sample does not provide a large increase in results. We start at a sample size of 100,000 posts, and increase the number of posts by 50,000 until we are satisfied with the resulting samples.

Figures 3.3 and 3.4 give the results with standard FolkRank for each of the examined training sample sizes on the Citelike and Delicious datasets respectively. The left side shows the recall graph of each of the individual sample runs, where runs of the same sample size are plotted in the same line style. The box plot on the right gives the average recall@5 per sample size. The more we increase the sample size, the less variation in results on samples of the same size, and the improvement in average recall is also smaller. As an outcome of this process we have found that a sample size of (roughly) 250,000 posts gives acceptable results. For BibSonomy

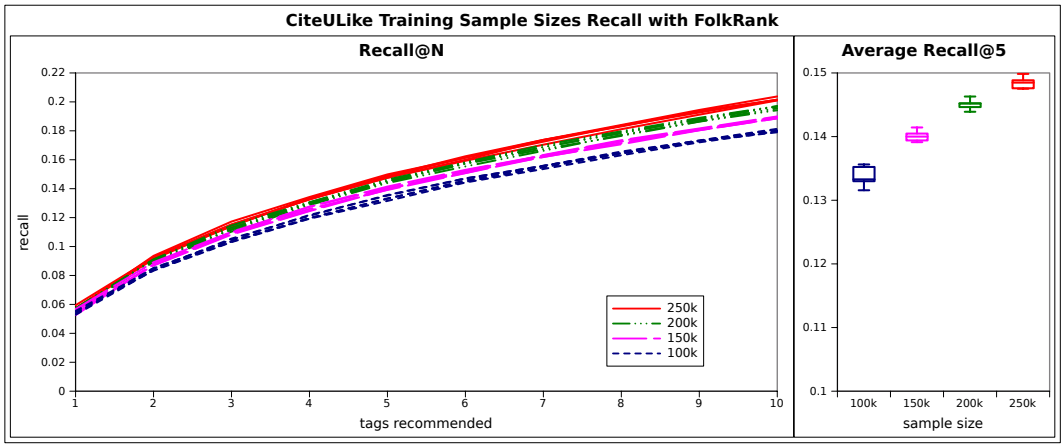


Figure 3.3: Recall with FolkRank for Training Sample Sizes of CiteULike

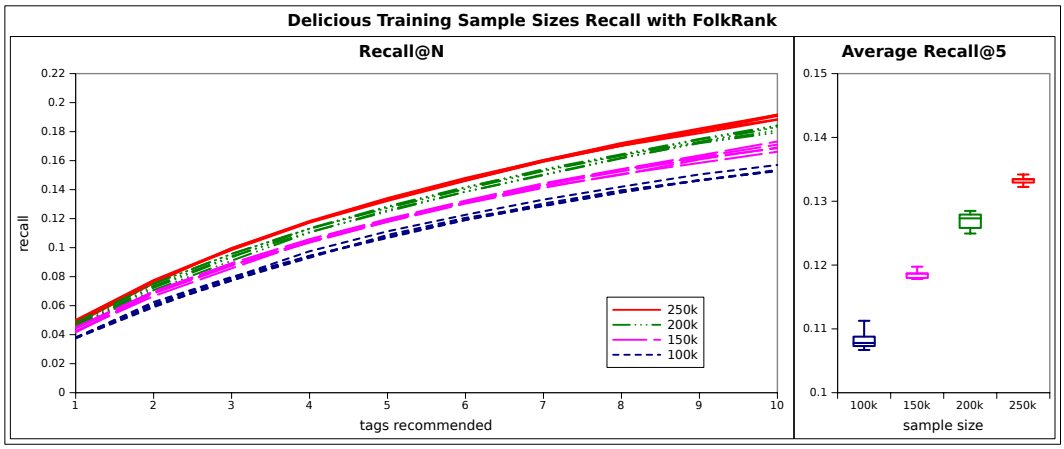


Figure 3.4: Recall with FolkRank for Training Sample Sizes of Delicious

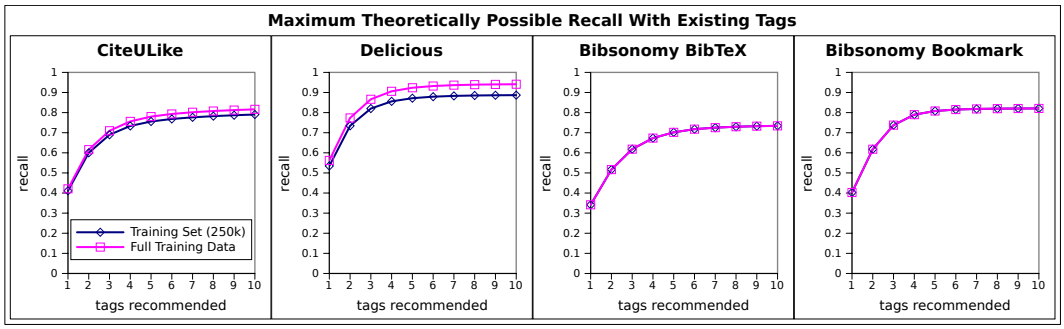


Figure 3.5: Theoretical Maximum Recall Achievable With Existing Tags

Bookmark and BibSonomy BibTeX we do not sample the training data and use all posts from the year previous to the test time-frame, as the number of posts in these datasets is sufficiently small. The statistics of the final training and test sets used in our experiments are given in Table 3.1. The sampling does have the effect that some of the tags in the test set of CiteULike and Delicious will not be present in their respective training sets, and thus cannot be recommended successfully by any of the evaluated recommenders. In Figure 3.5 we show the theoretical maximum possible recall that could be achieved on the test set of each dataset with recommending tags that exist in the training set. The theoretical maximum is calculated by assuming that only correct tags are recommended from the set of all tags that can be found in the training data. The maximum possible recall for a test post when recommending N tags is then given by $max\_recall@N = \frac{L}{\mathrm{TP+FN}}$, where L is the minimum of N and the number of true tags for the test post which also exist in the training data, and the denominator is the size of the complete true true tag set of the post. Figure 3.5 shows the maximum recall for our training set sample as well as the full training data. The extent of the problem of not including all training data tags in the samples is not too great, and we do not believe that this will impact the validity of our conclusions as all of the evaluated recommenders will suffer from this problem to the same degree. In addition to the training-test split, we create a separate evaluation split for each dataset that we use for comparison of individual approaches and for parameter tuning. The evaluation test and training sets are created from the datasets prior to the two months of real test data, in the same fashion as described above.

	Training Set (250k Sample)				
	Posts	Tag Assignments	Users	Docs	Tags
CiteULike	249,968	1,148,011	12,908	218,601	138,024
Delicious	253,890	566,173	30,848	109,201	56,338
BibS. Bookmark	42,325	179,599	982	40,679	24,830
BibS. BibTeX	17,560	66,529	1,264	16,360	17,307
	Test Set				
	Posts	Tag Assignments	Users	Docs	Tags
CiteULike	76,491	301,779	4,930	69,161	52,576
Delicious	1,746,483	4,431,116	43,997	877,593	175,146
BibS. Bookmark	9,506	30,053	243	9,425	4,811
BibS. BibTeX	2,843	10,657	333	2,746	3,943

Table 3.1: Training and Test Set Sizes (No-Core)

3.4.5 Training and Test Set for Post-Cores Level 2

We also evaluate our approaches on each of the datasets at post-core level 2. For each dataset, we create the post-core by iteratively removing posts where the user, document or one of the tags does not satisfy the condition that they appear in at least two posts. We then use a leave-one-out per user split to create the training and test sets by selecting the most recent post for each user and placing it in the test set. For parameter tuning we create an additional evaluation split from the resulting training data.

3.5 Evaluation and Results

In our evaluation we aim to first identify the best approach of including content, and then measure the overall impact of including content into the recommendation process. To answer the research questions below and find the best content inclusion method, we run experiments on the evaluation set with default parameter values. The default value of the dampening factor in our FolkRank approaches is $d = 0.5$, and the balance in weight between the query user and query document is set to $b = 0.5$ for all recommenders. We then tune d and b on the evaluation set, and finally give results on the test set with tuned parameters.

- How should content be included: directly into the model at the word level or indirectly at the document level?
- What is the most predictive source of content: document title or fulltext/ abstract?
- How much content should be included?

3.5.1 Baselines Without Content

Before including content, we first examine the prediction accuracy of the solely folksonomy-based recommenders without content. Figure 3.6 shows the recall@N achieved with the individual parts of our co-occurrence recommender and the final CoOcc recommendation set. On all datasets User Tags (UT) performs reasonably well, while Document Tags (DT) gives very low values of recall. Since most of the query documents have not been previously tagged, DT cannot recommend any tags in the majority of cases which leads to the low results. Considering the query user as well as the document by combining User Tags and Document Tags (UT∪DT)

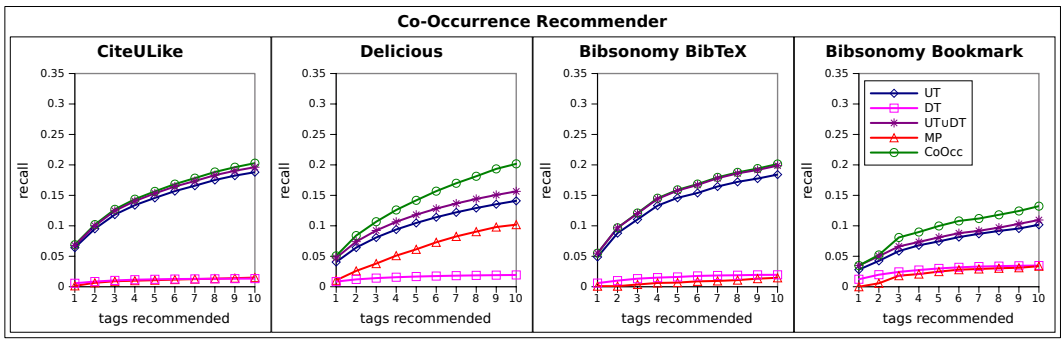


Figure 3.6: Co-Occurrence Recommender Parts and Final Recommendation Set

gives better results than only considering the query user. An interesting observation is that Most Popular Tags (MP) gives relatively high results on the Delicious dataset. We suspect that this is due to the biased crawling method used to obtain the Delicious dataset as discussed in Section 2.6.1. The difference between CoOcc and UT∪DT is that the recommendation set of CoOcc includes tags from MP for the cases where UT∪DT could not recommend any or the wanted number of tags. While on CiteULike and BibSonomy BibTex there is not much difference in results between CoOcc and UT∪DT, filling the recommendation set with tags from MP provides a significant improvement in results for Delicious and BibSonomy Bookmark.

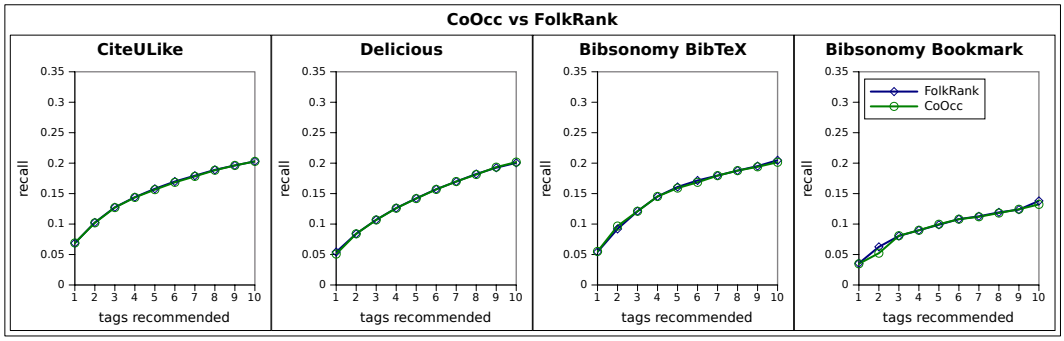


Figure 3.7: CoOcc vs FolkRank

When comparing the accuracy of the CoOcc recommender and plain FolkRank (Figure 3.7) both give almost equivalent results on all datasets. This is surprising since our intuition was that the graph-based FolkRank should achieve a higher prediction accuracy than the simpler CoOcc recommender. FolkRank has the ability to analyse the full folksonomy graph when generating recommendations, while CoOcc only considers the immediate neighbourhood of the query user and query

document which would correspond to one hop in the folksonomy graph. FolkRank has much more information on which to base the recommendations than CoOcc, however, they produce equivalent results. While the focus of this chapter is on the inclusion of content data into the recommendation process, we analyse why this might be the case in Chapter 4.

3.5.2 Direct vs. Indirect Content Inclusion

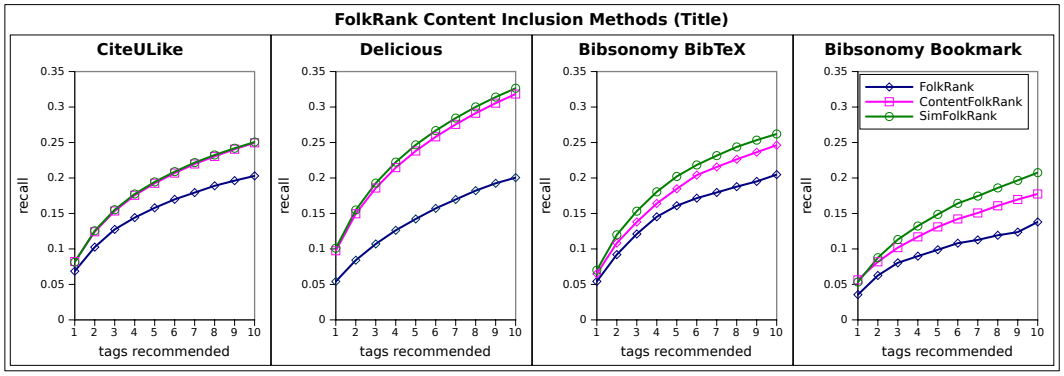


Figure 3.8: ContentFolkRank vs. SimFolkRank

In Figure 3.8 we show the results of evaluating our two methods for including content into FolkRank. On all datasets, the inclusion of content data gives a significant improvement over plain FolkRank. Comparing the two approaches, the indirect content inclusion method of adding similar documents to the preference vector (SimFolkRank) gives better results than incorporating the document content directly into the graph (ContentFolkRank). The biggest difference is on the BibSonomy datasets, while for CiteULike the results are almost identical with SimFolkRank performing slightly better. We assume that ContentFolkRank gives worse results due to the word nodes in the graph being connected to many more tags compared to the document nodes in the standard folksonomy graph used by SimFolkRank. The same individual word can appear in a variety of documents from different domains and thus be connected to many tags which are themselves unrelated. To accurately capture the query document several words are required in combination. The predictions generated by ContentFolkRank can be influenced by the edge configuration of individual words, which might be most connected to tags from a different domain than the query document whilst being connected to appropriate tags with less edge weight. In SimFolkRank, the similarities to training documents are calculated based on the whole representation of the query document,

and in the graph each of the similar documents is likely to be connected to tags from one or a few domains only. In the larger datasets of CiteULike and Delicious the difference between the two approaches is smaller. ContentFolkRank comes close in results to SimFolkRank, but does not outperform it. This suggest that with more data the weighting methods used in ContentFolkRank, which are based on Tf-Idf scores and include a co-occurrence element, can more accurately model the query document as well as the edge weights of words in the graph. With sufficient data the outcome of ContentFolkRank is thus very similar to SimFolkRank. However, in addition to producing better results, SimFolkRank is also computationally less expensive than ContentFolkRank since the ContentFolkRank graph is much larger due to the many word nodes.

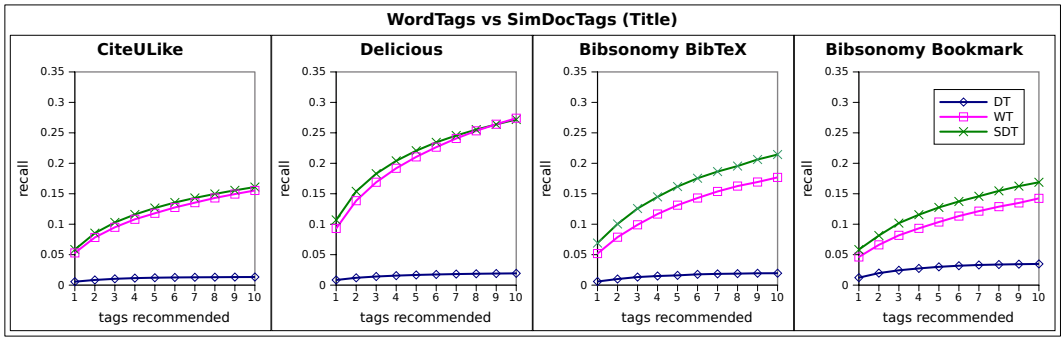


Figure 3.9: Word Tags vs. Similar Document Tags

Our co-occurrence recommenders with content are ContentCoOcc and SimCoOcc, where the difference between them is the content inclusion method used. ContentCoOcc uses Word Tags (WT) and SimCoOcc uses Similar Document Tags (SDT). Since the user-related part is the same in both recommenders, we can compare WT and SDT directly to ensure that no other factors influence the recommendations, and to ensure that our conclusions from the results of the content-aware FolkRank recommenders are valid. The results in Figure 3.9 confirm that including content via document similarity produces more accurate recommendations than including content directly by using word-tag co-occurrence.

3.5.3 Content Sources

To compare the prediction accuracy of the two content sources, we evaluate SimFolkRank and SDT using the title and the fulltext content of documents. For CiteULike and BibSonomy BibTeX we treat the abstracts of the bookmarked publications as the fulltext content. Figure 3.10 shows the results with SimFolkRank, and Figure

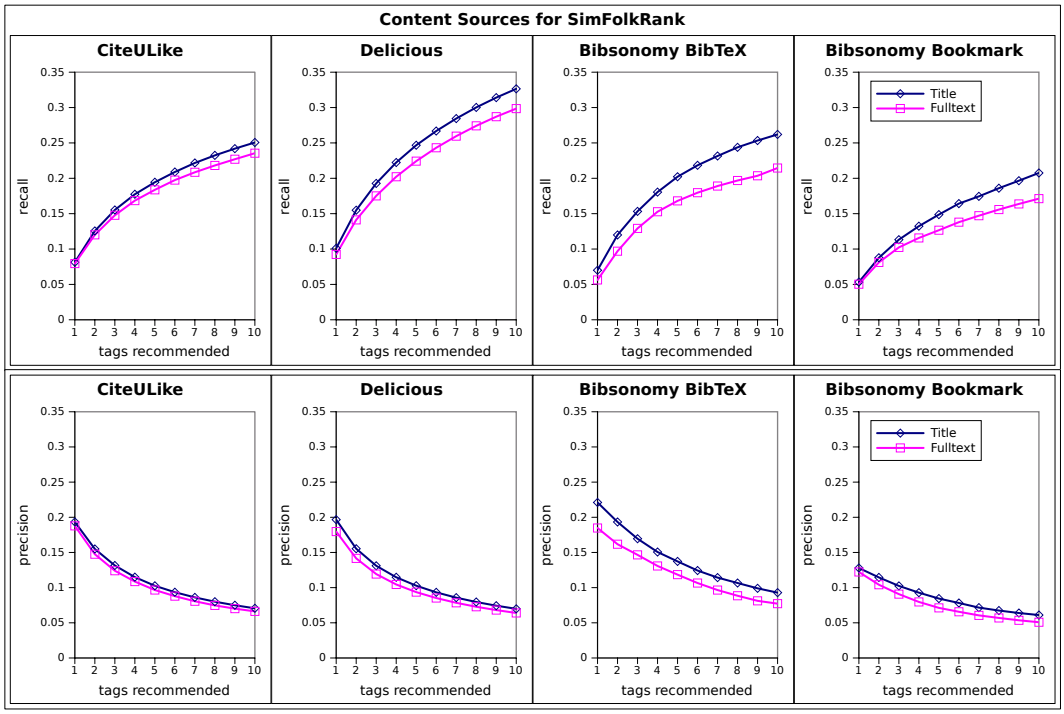


Figure 3.10: Content Sources for SimFolkRank: Title vs. Abstract/Fulltext

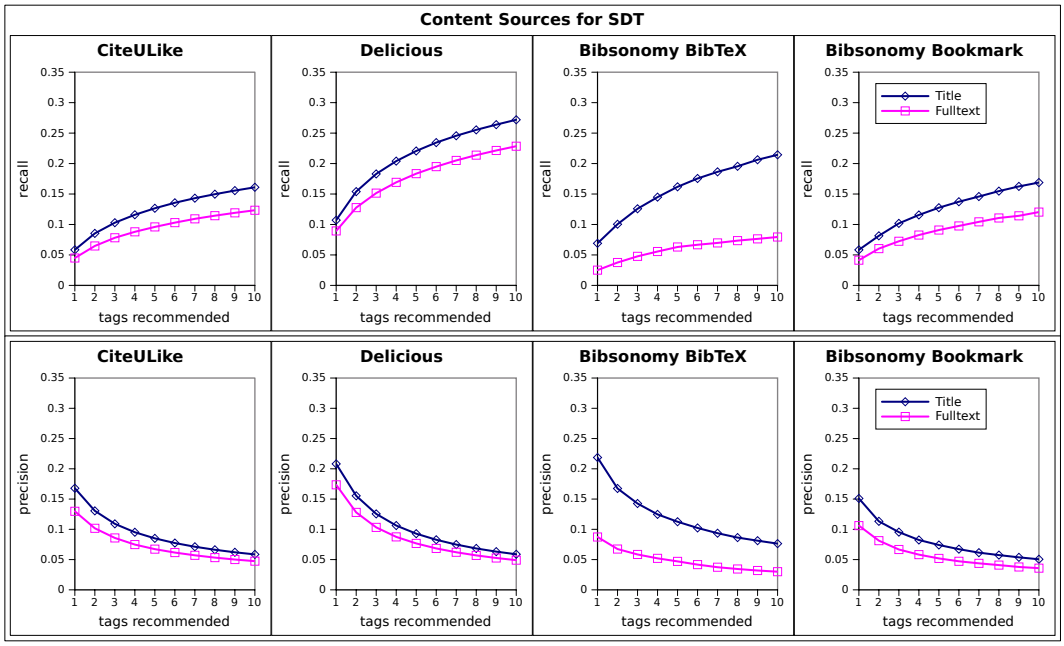


Figure 3.11: Content Sources for SDT: Title vs. Abstract/Fulltext

3.11 shows the results of the SDT part of the SimCoOcc recommender (without the influence of user-related tags). We compare the recall as well as the precision achieved with the two content sources. The usefulness of the title has been shown in [Lipczak and Milios, 2010b] with tag extraction, where words from the query document's title were selected and recommended as tags. The conclusion was that the title is a precise but limited content source, producing high precision but low recall. In our experiments we use the title to represent documents (in order to calculate document similarity), instead of directly recommending the title words as tags. The content-based tags recommended by our algorithms are thus not limited to tags that explicitly appear as words in the query document's title, but consist of tags that are related to the content of the document. Comparing the title and the fulltext as content sources in our approach, the title gives a higher precision as well as a higher recall across all datasets. Our results indicate that the title provides a more accurate representation of documents in the collection. Nevertheless, the fulltext data could still hold some value if used in addition to the title. In the future it would be interesting to investigate whether the two content sources can be successfully combined and results can be increased over using the title only.

3.5.4 Content Amount

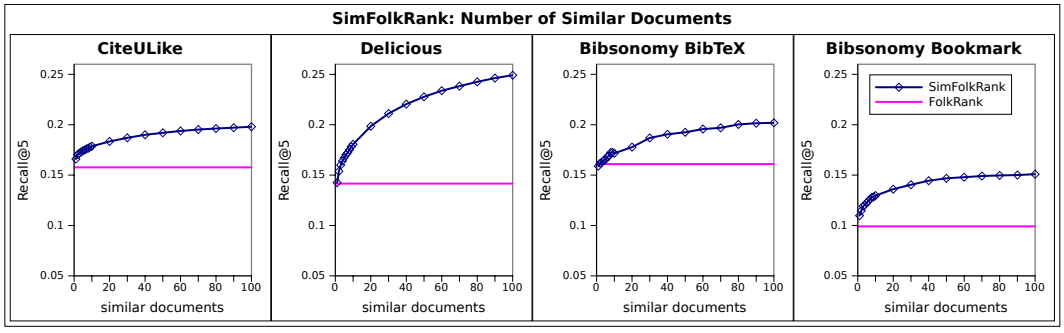


Figure 3.12: Content Amount in SimFolkRank: Number of Similar Documents

To evaluate the impact of the amount of content that is included we show the results of SimFolkRank (Figure 3.12) and SDT (Figure 3.13) with different numbers of similar documents considered. The content source in these experiments is the document title. The x-axis gives the number of most similar documents included and the y-axis is recall when recommending five tags. The horizontal line in each graph gives the recall@5 without including content. The results indicate that prediction results improve the more content is added, where the biggest gain is achieved by

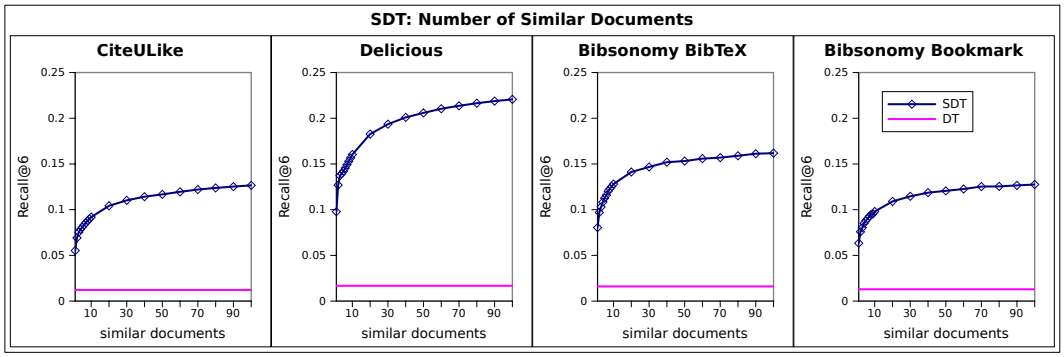


Figure 3.13: Content Amount in SDT: Number of Similar Documents

the most similar documents. The shape of the plots and the fact that the results do not decrease at higher numbers of similar documents also confirms that normalised cosine similarity is an appropriate metric for measuring document similarity in our scenario.

3.5.5 Parameter Tuning

In the following subsections we present the results of tuning the remaining parameters of our recommenders. The evaluation metric we use to identify the best parameter settings is recall@5.

Dampening Factor in FolkRank and SimFolkRank

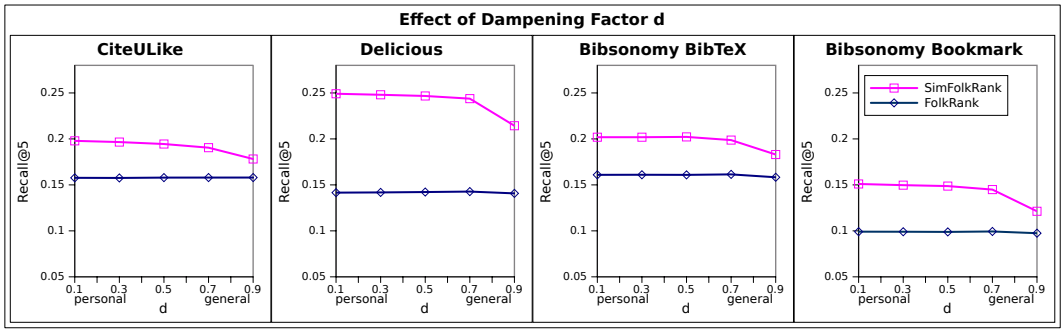


Figure 3.14: Tuning of Dampening Factor d

The dampening factor in FolkRank and SimFolkRank determines the importance given to personalised and general node weights during the weight spreading computation. With lower settings most of the importance is given to personalised

weights while at higher settings of d the general node weights also have an impact. In our parameter tuning runs (Figure 3.14), we have found a setting of $d = 0.1$ to give the best results for all datasets. An in-depth analysis and discussion of the impact of the dampening factor is presented in Chapter 4.

Balance Between Query User and Query Document

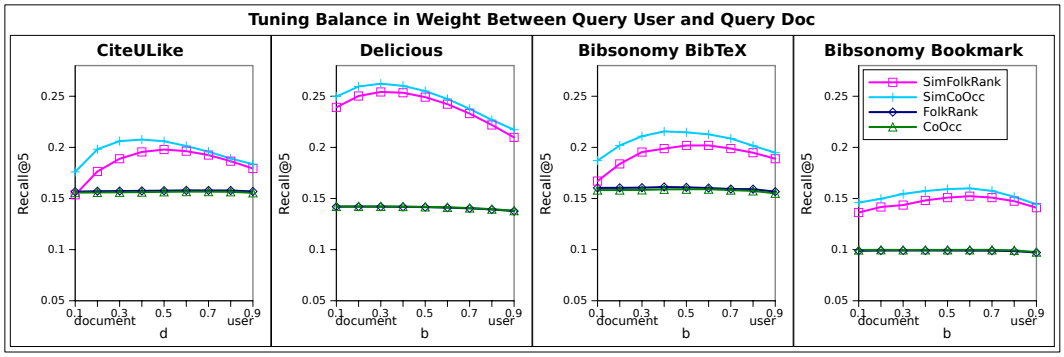


Figure 3.15: Tuning of Balance b Between Query User and Query Document

All of the examined recommenders have the parameter b which determines the balance in importance given to tag scores from the user side and the document side of the query. In co-occurrence approaches b is the combination weight between the user-related and the document-related recommendation sets. In our FolkRank approaches, b determines how the total preference weight is divided between query's user and document nodes. In the original FolkRank algorithm, b is not included explicitly as a parameter and is always implicitly set to a value determined by the training data. In original FolkRank $b = \frac{|U|}{|U|+|D|}$, which corresponds to setting b equal to the fraction of user-nodes in the graph. We have introduced b as a parameter in FolkRank and SimFolkRank to set the balance in preference weight independently of the number of user and document nodes in the training data graph.

In Figure 3.15 we present the results with different settings of b. Without including content (FolkRank, CoOcc), changing the value of b does not have a significant impact. Since most of the query documents in the test sets are new, the recommendations generated without including content will be from the user side only in the majority of cases. However, with content data (SimFolkRank, SimCoOcc) the recommendations are generated from both the user side and the document content, and we can clearly observe the impact of b. Surprisingly, the simple SimCoOcc recommender produce better results than SimFolkRank across all settings of b. For

the FolkRank approaches the results confirm that there is value in introducing the parameter b to explicitly set the balance between user and document side instead of using the strategy of the original FolkRank algorithm. Setting $b = \frac{|U|}{|U|+|D|}$ would results in values lower than 0.1 for all of the datasets except Delicious where it would be 0.2. For SimFolkRank the best results are achieved with setting b to 0.5 for CiteULike, 0.3 for Delicious, 0.5 for Bibsonomy BibTeX, and 0.6 for BibSonomy Bookmark. The best settings for SimCoOcc are 0.4 for CiteULike, 0.3 for Delicious, 0.4 for Bibsonomy BibTeX and 0.6 for BibSonomy Bookmark.

3.5.6 Results on Test Set

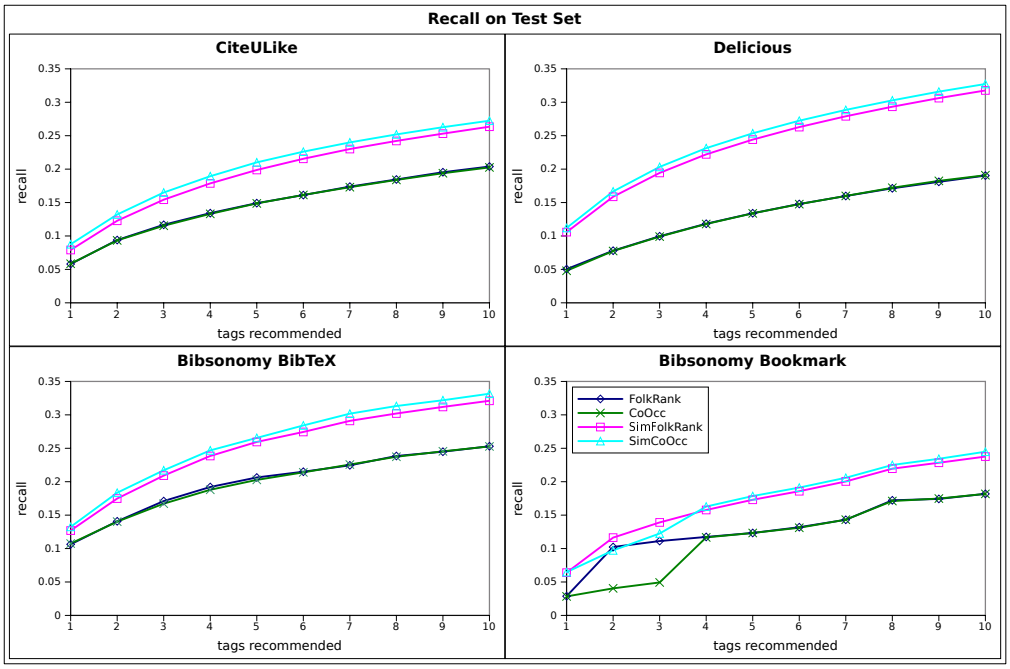


Figure 3.16: Recall on Test Set with Tuned Parameters

Here we present our final results with tuned parameters on the test set of each of the datasets. The content source in the content-aware approaches is the document title. For FolkRank approaches the dampening factor is set to $d = 0.1$. For all approaches the balance b in preference weight is set per dataset to the best value that was found in the parameter tuning runs. Figures 3.16 and 3.17 show the recall and F1 respectively, on the test set for each of the datasets. Including content into the recommendation process provides a significant increase in results. An interesting result is that the simpler SimCoOcc recommender produces better results than the

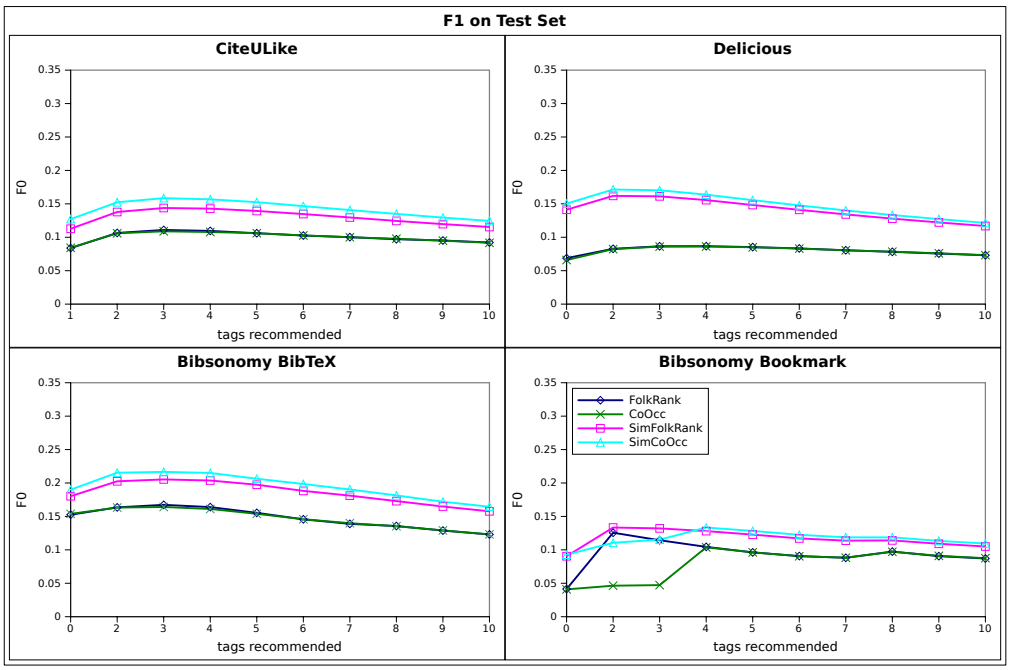


Figure 3.17: F1 on Test Set with Tuned Parameters

graph-based SimFolkRank across all datasets. SimFolkRank has more information to base its predictions on since it includes the full folksonomy graph in its tag score computation, and our intuition was that it should be able to leverage the complex connections in the graph to produce more accurate recommendations. However, this does not seem to be the case and the simpler SimCooc approach produces better results while being computationally much less expensive. The results on BibSonomy Bookmark without including content data (FolkRank, CoOcc) are due to the fact that a large portion of the test posts in BibSonomy Bookmark contain new users as well as new documents. For these test posts the algorithms which do not include content data default to recommending the overall highly-ranked tags in the graph without personalisation. FolkRank and CoOcc have different rankings for the top three tags in the general/most popular recommendations which leads to the difference in results when recommending up to three tags.

Post-Core 2

Figures 3.18 and 3.19 show the recall and F1 of our approaches on each of the datasets' post-core at level 2. The parameters are tuned on the post-core 2 evaluation set and the results shown here are with optimal parameter settings. For

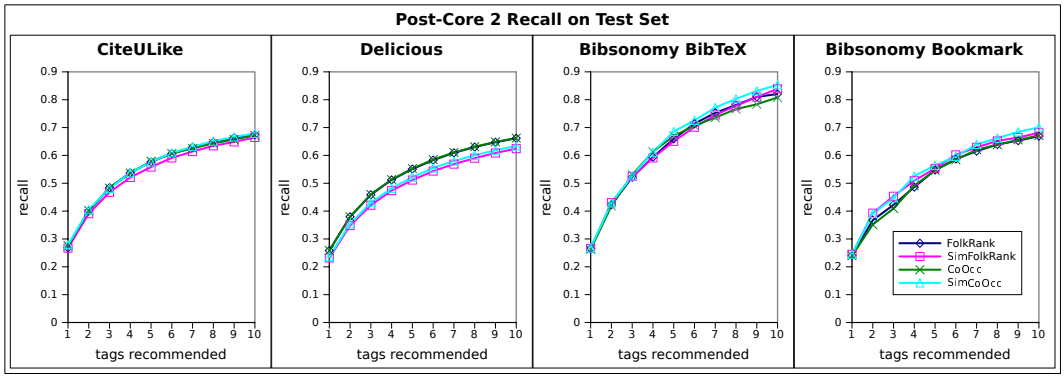


Figure 3.18: Post-Core 2 Recall on Test Set with Tuned Parameters

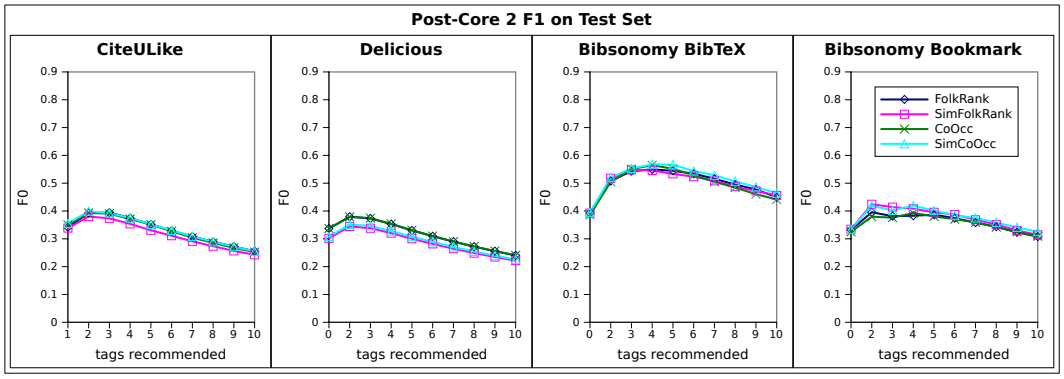


Figure 3.19: Post-Core 2 F1 on Test Set with Tuned Parameters

post-core 2 datasets, there seems to be no clear benefit in including content data. While on Bibsonomy BibTex and Bibsonomy Bookmark results are increased by including content, for Delicious both of the content-aware recommenders produce worse results. Since (almost) all of the documents in the test set for post-core 2 also exist in the training data, they all have previously assigned tags available which can be recommended. There seems to be no need to additionally include similar documents in the preference vector as well since the exact query document exists in the training data. Adding the content in this case does not provide a clear improvement and has a negative effect in some cases. This is an interesting result and suggests that the best strategy for the future might be to only include the content if the query document does not exist in the training data, for post-core 2 as well as unpruned datasets. On post-core 2 the co-occurrence recommenders perform slightly better than FolkRank approaches, with FolkRank giving equivalent but not better prediction accuracy in some cases.

3.6 Conclusion

We have shown that including content in the recommendation process of folksonomy-based approaches addresses the new item problem and significantly increases results on full tagging datasets. For datasets at post-core level 2, which do not exhibit the new item problem, there seems to be no benefit in including content as the exact query document already has tags assigned to it which can be recommended from the training data. The best content source and document representation in our experiments was the title of documents which gave better results than utilising the fulltext content. Furthermore, we have observed that including content indirectly by using a content-based document similarity measure produced better results than breaking up documents into their individual words and trying to exploit individual word-tag co-occurrence. Another interesting observation is that the more complex graph-based recommenders FolkRank and SimFolkRank do not provide a benefit over the simpler co-occurrence approaches CoOcc and SimCoOcc which are computationally much less expensive. CoOcc and FolkRank give nearly equivalent results while SimCoOcc actually produces more accurate tag predictions than SimFolkRank. This is a surprising result which suggests that there might be issues in the FolkRank approaches which hinder their ability to leverage the full folksonomy graph as an information source. We suspect the issues could be due to the underlying graph model used or due to the weight spreading algorithm of FolkRank. We explore these two topics in the next chapter. In summary, the conclusions of this chapter are the following.

- Including content into the recommendation process addresses the new document problem and significantly increases results on full/unpruned datasets.
- The title of documents is a better content source than the full-text.
- Including content at the document level produces a more accurate recommender than including content at the word level.
- The graph-based FolkRank algorithm does not provide a benefit over simpler co-occurrence recommenders.

Chapter 4

Graph-Based Ranking

In this chapter we present an in-depth analysis of the inner workings of FolkRank, highlight issues which might reduce tag recommendation accuracy, and propose novel adaptations to overcome these. As we have shown in the last chapter, FolkRank did not produce a higher prediction accuracy than the simple CoOcc recommender despite being able to consider the full folksonomy graph and thus utilise a much larger information scope when generating predictions. In the following sections we aim to investigate why this was the case and propose methods to potentially improve FolkRank's predictions accuracy. The first part of our analysis concerns itself with the folksonomy graph used in FolkRank to model the tagging data. We highlight information that is lost and implicit assumptions that are made by the model and propose a novel graph structure which captures the tagging data more accurately. In the second part we conduct a detailed analysis of FolkRank's iterative weight spreading algorithm and identify issues that exist therein. To address these issues we present a novel weight spreading approach which we call PathRank. Even though using PathRank does not significantly improve the prediction accuracy in our experiments, it allows for a more comprehensive analysis of weight spreading in social tagging data and is also computationally much less expensive than FolkRank. Finally, we provide an extensive theoretical discussion as well as practical evaluation of the value of exploring the deep folksonomy graph. We evaluate whether the potential benefit of considering the information contained in deeper levels of the graph is worth the added computational expense and present important insights regarding the applicability of deep graph-based methods to social tagging data in general. In summary, our main contributions are:

- An improved graph data model which more accurately captures the tagging data.

- An alternative weight spreading algorithm for social tagging data which is more traceable and less expensive than FolkRank's iterative approach.
- An in-depth theoretical discussion as well as practical evaluation of the value of exploring the deeper folksonomy graph for tag recommendation, and of the applicability of graph-based methods to the domain of social tagging in general.

4.1 Graph Models and Edge Weights

The first issue we examine is the graph structure of the folksonomy model and the problem of setting the edge weights. As described before, a folksonomy is a tuple (U, D, T, A) where U is the set of users, D is the set of documents, T is the set of tags and $A \subseteq U \times D \times T$ is the set of tag assignments. A tag assignment $a \in A$ is a triplet (u, d, t) and indicates that user u has assigned tag t to document d. A post in the tagging data consists of a set of tags assigned by a user to a document. Posts themselves are only captured implicitly in the folksonomy model. The set of posts can be described as $P \subseteq U \times D \times 2^T$ where 2^T is the power set of T and each post $p \in P$ is a triplet (u, d, S) consisting of a user $u \in U$, a document $d \in D$, and a set of tags $S \in 2^T$. Due to the fact that a post can contain a variable number of tags and since the post-membership information of tag nodes is not included explicitly in the folksonomy model, the user and document nodes in the graph can be connected to a variable number of tag nodes. The variable number of tags per post affects the outcome of weight spreading since in each spreading action the weight that is passed to each connected node depends on the total number and weight of edges of the active node. The difficulty is then setting the edge weights in the graph, where each alternative method of doing so makes different assumptions. In the following paragraphs we explore alternative graph construction methods and the implicit assumptions they make. We later evaluate each of these methods in Section 4.3. An assumption which holds in all alternatives is that the co-occurrence of users and tags, as well as documents and tags, should influence edge weights. The weight of a user-tag edge should be higher if the user has used the tag multiple times to tag multiple documents. Similarly, if the same tag has been assigned to a document multiple times, so by multiple different users, the document-tag edge should be given a higher weight. In contrast, for user-document relationships the tagging data only provides one distinct co-occurrence since a user can only tag each document once (with a set of tags).

4.1.1 Folksonomy Graph

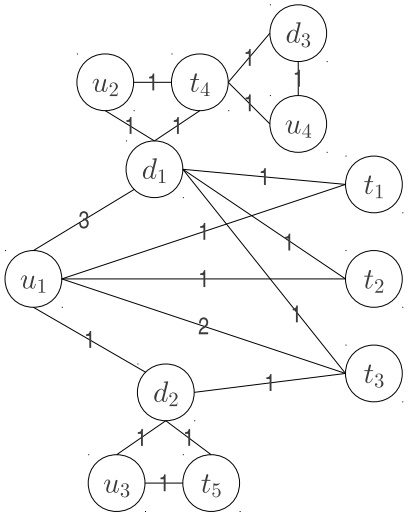


Figure 4.1: Folksonomy Graph

Based on the folksonomy model, FolkRank creates an undirected tri-partite graph $G = (V, E)$ where users, documents and tags are represented as nodes $v \in V$, and all co-occurrences of users and documents, users and tags, and documents and tags are edges $e \in E$ between the corresponding nodes. The weight of the edge between two nodes depends on the number of their co-occurrences, given as the number of tag assignments that both nodes appear in. The folksonomy graph structure and edge weighting methodology used in original FolkRank is given in Figure 4.1. User u_1 has tagged document d_1 with tags t_1, t_2 and t_3, and has tagged document d_2 with just t_3. In the folksonomy graph, the weight between user and document nodes is set according to the number of tag assignments, and thus the number of tags in the post. Thus the weight of the edge u_1-d_1 is equal to three while the weight of edge u_1-d_2 is equal to one. This means that within the context of a post, all types of nodes (user, document, and all tags together) get the same amount of total weight. In the context of post $(u_1, d_1, [t_1, t_2, t_3])$ only, ignoring the influence of post $(u_1, d_2, [t_3])$, the weight of the edge u_1-d_1 is the same as the sum of edge weights u_1-t_1, u_1-t_2 and u_1-t_3, which is 3. However, another consequence of this graph construction method is that, if we spread weight from u_1, then d_1 would get a higher weight than d_2, and subsequently, the tags connected to d_1, in this case t_4, would get a higher weight than the tags connected to d_2, namely t_5. The implicit assumption made by this model is that documents to which a user has assigned many tags are more representative of the user's interest. Another assumption is

made with regard to the number of tags in a post. If we had a query post $(u_1, d_3, \emptyset)$, the fraction of weight spread from u_1 to t_3, which is the user's most used tag, would be 2/8 (times the dampening factor). However, the query document d_3 would spread 1/2 of its weight to t_4. Assuming both the query user and query document have the same preference weight, t_4 would thus be ranked higher than t_3 even though t_3 has been used by the user multiple times and t_4 has only been assigned to d_3 once. The assumption which leads to this outcome is that if a post has multiple tags then each of the tags is proportionally less important to the user and document of the post. While these assumptions are not wrong as such, they are implicitly made by the graph model and influence the outcome of weight spreading. We believe it is important to investigate and highlight the impact of making these implicit assumptions on the outcome of weight spreading, and thus prediction accuracy of the recommenders. The described assumptions made due to the graph model when spreading weight in the folksonomy graph can be summarised as follows.

- Within the context of a post, all types of nodes (user, document, tag) have the same amount of relevance summed by node type.
- The weight of the user-document relationship depends on the number of tags in the respective post. The more tags a user has assigned to the document, the stronger the user-document connection.
- Each tag in a post is proportionally less important to the user and document if the post contains multiple tags.

4.1.2 Folksonomy Adapted Graph

We propose an alternative edge weighting method for the folksonomy graph, illustrated in Figure 4.2, which we refer to in our experiments as the Adapted Graph (AG). The difference to the original folksonomy methodology is that we always keep user-document edges at a weight of 1 regardless of the number of tags in the post. However, this also means that in our example of spreading weight from user u_1, the sum of the weights spread to tag nodes t_1, t_2 and t_3 would be higher than the sum of the weights spread to document nodes d_1 and d_2. By spreading weight in the Adapted Graph, the following assumptions are made.

- Within the context of a post, all tag nodes together are more important than user or document nodes.
- The weight of the user-document relationship is independent of the number of

tags in the respective post. All edges connecting users to documents have the same weight.

- Each tag in a post is proportionally less important to the user and document if the post contains multiple tags.

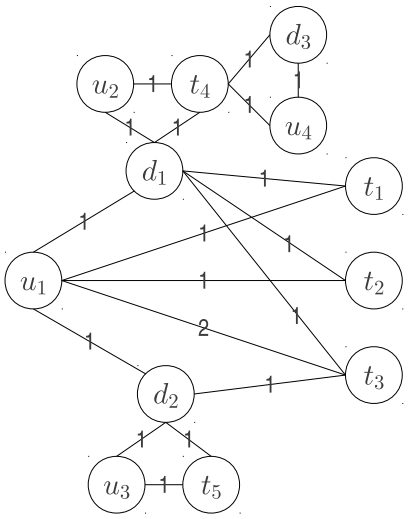


Figure 4.2: Folksonomy Adapted Graph

4.1.3 Post Graph

Since both aforementioned graph construction methods do not explicitly include the post-membership information of nodes, we believe that they produce an inaccurate model of the social tagging data, and propose a structurally different graph model which we call Post Graph (PG). The Post Graph model includes an additional type of node representing posts themselves into the graph. Figure 4.3 shows the Post Graph for the same data as the previous folksonomy graph and Adapted Graph models. The user, document and tag nodes are only connected to post nodes instead of being directly connected to each other. Furthermore, we set the weight of post-tag edges so that the edge weights to all tags of a single post sum to one. The sum of edge weights for each post is thus equal to three, where the post-user edge has a weight of one, the post-document edge has a weight of one, and all of the post-tag edges together also have a total weight of one. This makes the strength of the user-document relationships independent of the number of tags in the post, as well as ensuring that the same amount of total weight is spread to all types of nodes in the context of a post. To address the assumption that having multiple tags in a

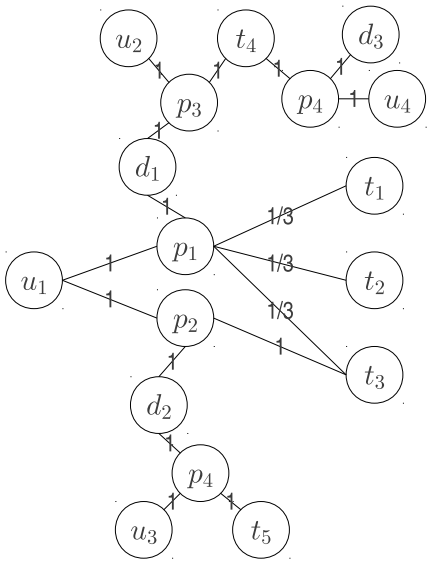


Figure 4.3: Post Graph

post implies less importance for each of them we evaluate an alternative method of retrieving tag scores from the graph. Instead of directly retrieving tag scores as the weight of tag nodes, we retrieve the weight of post nodes instead, and in a second step calculate the tag scores by summing up for each tag the weight of the post nodes that the tag is related to (ignoring the total number of tags in each post). For example in Figure 4.3, the final score for tag t_1 would be equal to the weight of the post node p_1. The number of other tags that are also connected to p_1 (in this case t_2 and t_3) would not have an impact on the score of t_1. For t_3 the final score would be calculated as the sum of the weights of p_1 and p_2 since t_3 is connected to both of these post nodes.

- Within the context of a post, all types of nodes (user, document, tag) have the same amount of relevance summed by node type.
- The weight of the user-document relationship is independent of the number of tags in the respective post. All relationships between user and document nodes (via post nodes) are of equal strength.
- By retrieving each tag's score as the sum of weights of post nodes it is connected to, the importance of each tag is independent of the total number of tags in its post.

Additionally, if we compare the Post Graph model to the hypergraph model [Symeonidis et al., 2008], the Post Graph captures all of the information contained in the hypergraph without information loss or aggregation. Since all of the user, document and tag nodes are only connected to post nodes and not directly to each other, a relationship between for example a user u_1 and a tag t_1 is always complemented by the corresponding document d_1 (via post p_1). User u_1 cannot be directly connected to tag t_1 as is the case in the folksonomy graph and Adapted Graph models, and thus no aggregation of information is done when constructing the Post Graph from the tagging data. Furthermore, the Post Graph captures even more information than the hypergraph, by explicitly including post nodes. The Post Graph model captures not only the three-dimensional relationship (or tag assignment) between a user u_1, document d_1 and tag t_1, but also the fact that two other tag assignments (u_1, d_1, t_2) and (u_1, d_1, t_3) are part of the same post p_1.

4.2 Weight Spreading and Value of Deep Graph

FolkRank's iterative weight spreading algorithm has two potential advantages over approaches which only utilise the immediate neighbourhood of the query nodes, such as simple co-occurrence methods. Firstly, the general importance of tags is taken into account when generating recommendations. All nodes are initialised with random starting weights, and the weights are then redistributed among nodes during the iterative weight spreading calculation. The general importance scores of tags can also be described as authority-based popularity, due to the characteristic that important user or document nodes will provide more weight to their connected tags. Secondly, the weight spreading algorithm considers the information contained in the deep graph. In Figure 4.4 we illustrate how FolkRank utilises the deeper graph beyond the immediate neighbourhood of the query user and document. User u_1 and document d_3 make up the query post, the immediate neighbourhood of the query nodes is shown in solid lines, the deeper graph is shown in dashed lines, and the weights of all edges which are not explicitly labelled are set to 1. In approaches considering only the immediate neighbourhood of u_1 and d_3, the candidate tag set for this query post would consist of tags t_1, t_2, t_3, t_4, t_5, and t_6. A co-occurrence approach, such as our combination of user-related and document-related tags (CoOcc), would rank t_1 as the best recommendation as it is related to both u_1 and d_3, followed by t_2 as the second best since it has a relatively strong relationship with u_1. However, when trying to rank t_5 and t_6, both of these tags would have the same prediction score and the algorithm would not have sufficient information to decide

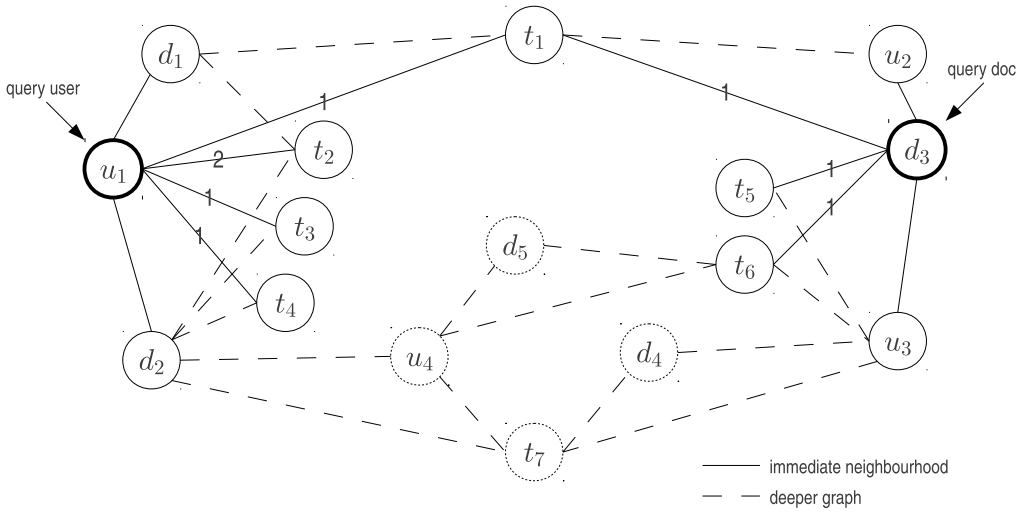


Figure 4.4: FolkRank Utilisation of Deep Graph

which of them should precede the other in the ranking. In the final tag predictions, the ordering of t_5 and t_6 would be random. By utilising the deeper graph, FolkRank's iterative weight spreading algorithm has the ability to provide a definitive ranking of t_5 and t_6 by trying to deduce which of them is more important to the query nodes. It would spread weight along the path $u_1 \rightarrow d_2 \rightarrow u_4 \rightarrow t_6$ and thus t_6 would be ranked higher than t_5. The other method that FolkRank has for breaking ties and re-ranking tags which would otherwise have equal prediction scores are the general importance weights. Additionally, FolkRank also spreads weight to tag t_7 found in the deeper graph and includes it in the candidate set, whereas t_7 would be omitted by approaches which only recommend tags co-occurring with the query user or document.

It seems intuitive from the graph structure and from literature applying graph-based approaches to non-folksonomy data that t_7 should receive some weight and be included in the candidate tag set, and that t_6 is more related to the query and thus should be ranked higher than t_5. However, the value of following this computationally expensive strategy and considering the connections of the deep folksonomy graph has not yet been directly evaluated. As there are other factors that impact the weight spreading calculation of FolkRank, which we explore in the following paragraphs, it has not yet been established that considering the deep graph provides an increase to tag recommendation accuracy. As part of the theoretical discussion, the opposite argument to FolkRank's assumption about the predictive indications of the deep graph could also be made. Considering the query user u_1,

if we make the somewhat weak assumption that u_1 is aware of the existence of tag t_7, then it would not make sense to spread weight to t_7. In the graph in Figure 4.4, user u_1 has tagged d_2 with the tags t_2, t_3 and t_4. A different user u_4 has also tagged the same document d_2 with tag t_7. If we assume that u_1 has, in his view, completely described d_2 with t_2, t_3 and t_4, this would suggest that t_7 was not required by the query user u_1 to describe d_2. One could thus argue that this was a conscious decision and t_7 might not be considered to be a good descriptor by u_1 in general. The weak points of this argument are the generalisation and the assumption of completeness. Rather than dismissing tag t_7 completely, u_1 might also think that t_7 is not appropriate for the documents he has tagged so far but generally a useful descriptor. More importantly, user u_1 might not be aware of tag t_7 at all. However, starting from the query document d_3, a similar and more convincing argument can be made with regard to t_7. Document d_3 has been tagged with t_5 and t_6 by user u_3, who has also tagged a different document d_4 with t_7. In this case the argument against assigning a higher weight to t_7 as a candidate tag for d_3 is much stronger. Since user u_3 has used t_7 for a different document (d_4), we can take this as an indication that he is aware of the tag's existence, and has explicitly not assigned t_7 to d_3. He is using the different tag sets of $\{t_5, t_6\}$ and $\{t_7\}$ to distinguish between documents d_3 and d_4. If any deduction is made about the relevance of t_7 to d_3, it should be that the graph indicates a negative relationship and the weight of t_7 with regard to d_3 should be reduced rather than increased. While this hypothesis is plausible, it still comes with some limitations which would be interesting to explore in the future. It assumes that the user has a complete recollection of all tags he has used in the past, and that he has decided to assign all of the tags from his tag vocabulary that would be appropriate. These assumptions might not hold in some cases. Since time is not captured and considered in the graph model, we cannot be sure that the user made a conscious decision against the use of a tag in cases where the tag has not been used by the user for some time. Instead of considering the tag as a candidate and dismissing it, it could be the case that the user has discontinued using the tag altogether out of personal preference or has just forgotten about it. In the future it would be worthwhile to explore a weighting method for the extracted negative relationships that takes time into account. A stronger negative relationship could then be calculated for tags which have been recently used by a user for other documents (but not the document in question), and a weaker or no negative relationship could be assigned to tags which have not been used in a long time.

The counter-argument to utilising even longer paths, which leads to Folk-

Rank's ranking of t_6 above t_5, is the highly personal tagging behaviour of users in (broad) folksonomies. FolkRank uses the path $u_1 \rightarrow d_2 \rightarrow u_4 \rightarrow t_6$ to deduce that t_6 is more relevant than t_5 to the query consisting of user u_1 and document d_3. However, this deduction is based on the fact that t_6 was used by a different user u_4 for a different document d_5, and the only link to the query is given by u_4 having tagged d_2 which has also been tagged by the query user u_1. The shared document d_2 is taken as an indication that u_1 and u_4 have similar interests and that u_1 should give some authority to all of the other opinions/tag assignments made by u_4. In [Wetzker et al., 2010], Wetzker et al. argue that tag assignments cannot be transferred as easily across users and provide evidence for the highly personalised tagging behaviour of users in broad folksonomies. They show that users who have tagged the same documents rarely assigned the same tags to these documents. Even though the users' areas of interest are similar due to the shared documents, only a small overlap can be observed in their tag vocabulary which indicates that the users' views of the documents are highly personal.

4.2.1 Analysis of Iterative Weight Spreading in Folksonomies

In the following paragraphs we analyse the iterative weight spreading method of FolkRank in detail and address issues which we believe to hinder or cascade its ability to effectively utilise the information contained in the deeper graph. An important preliminary observation about FolkRank is that the impact of each preference node on the final weights in the graph is independent of the influence of other preference nodes. Having multiple nodes with preference weight > 0 in the preference vector of FolkRank is virtually equivalent to performing the weight spreading computation for each of the preference nodes separately, and then doing a linear combination of the resulting weight vectors to give the final ranking. As long as the end condition of the weight-spreading iterations is set sufficiently small, the only nodes which could end up with a different weight are the ones at the very bottom of the ranking. This method can be used to speed up the prediction time of FolkRank in a live tag recommendation scenario. For each user in the system, the tag scores can be pre-calculated offline and stored. The same can be done for each document. During the online tag recommendation phase, the pre-calculated tag scores for the query user and query document would then be retrieved and a weighted average of the scores would be taken per tag in order to quickly create tag recommendations. For the following discussion we assume that this method of performing a separate weight spreading computation for each of the preference nodes is used, so that each individual weight spreading run will have only one node in the preference vector.

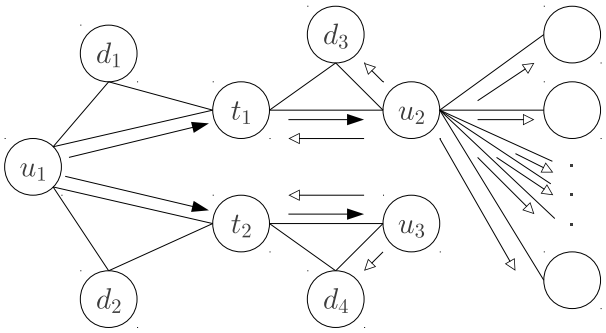


Figure 4.5: FolkRank Swash-Back Problem

Swash-Back Problem

A problem of FolkRank as discussed in [Jäschke et al., 2007] is "swash-back" of weights. Since the graph is undirected, weight is spread from a node n_1 to a connected node n_2 in one iteration and then spread back from n_2 to n_1 in the next iteration. This means that the weight of n_1, the node from which the weight-spreading originates, is adjusted in the second iteration based on the (number of and weight of) edges of n_2, which does not seem intuitive and is not desirable. We illustrate the consequences of this in Figure 4.5. User u_1 has tagged documents d_1 and d_2 with tags t_1 and t_2 respectively. Tag t_1 is also used by a very active user u_2 who is connected to a large number of other nodes, where t_2 is also used by user u_3 who has only one tag assignment. If we want to recommend tags for u_1 and activate that node in the graph, t_1 and t_2 would get the same weight in the first iteration. In the second iteration t_1 spreads weight to u_2, and t_2 spreads weight to u_3 (as well as all of their other connected nodes), where the weight received by u_2 and u_3 is equal. The third iteration is when the swash-back with regard to t_1 and t_2 occurs, denoted by the empty arrows. Tag t_2 gets half of the weight of u_3 (times the dampening factor) back since u_3 is connected to two nodes. However, u_2 is connected to many other nodes, and so the weight spread back from u_2 to t_1 would be much less than the weight spread back from u_3 to t_2. In the final tag predictions for user u_1, tag t_2 would have a higher score than t_1 due to the behaviour of users u_2 and u_3. This is contrary to our intuition that the weights should be equal up to this point since the query user u_1 has used both tags with equal frequency in the past. In the final ranking, when the node weights of the query user are combined with the node weights produced by the query document, the weights of t_1 and t_2 are expected to change to reflect the influence of the deeper graph. However, in this weight spreading operation for the query user only, the only source of preference weight is u_1. The change

in weights due to swash-back might outweigh the later influence of other preference nodes and prevent FolkRank from utilising the information contained in the deeper graph.

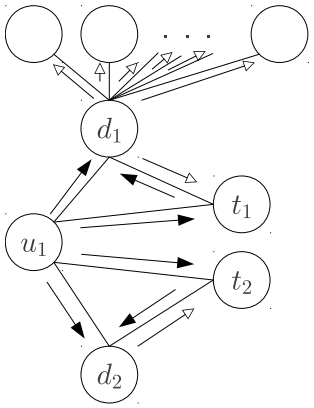


Figure 4.6: FolkRank Triangle Spreading Problem

Triangle-Spreading Problem

Another issue, which we refer to as triangle-spreading of weights, is illustrated in Figure 4.6. User u_1 has tagged document d_1 and d_2 with tags t_1 and t_2 respectively. Document d_1 is a popular document tagged by many other users, whereas d_2 has only been tagged by u_1. If we activate u_1 in order to recommend tags for this user, tags t_1 and t_2 would get the same weight in the first iteration. In the second iteration, d_2 would spread half of its weight to t_2 (times the dampening factor), however, d_1 would spread less of its weight to t_1 since d_1 is also connected to several other nodes. This would mean that in the tag weights for query user u_1, tag t_2 would get a higher weight than t_1 even though the user has used both tags with equal frequency. A similar problem would arise with regard to the weight of documents d_1 and d_2 if one of the tags was very popular. Due to graph being undirected and the folksonomy consisting of triplet relationships (user, document, tag), if two nodes n_1 and n_2 are connected, there is always at least one indirect path from n_1 to n_2 via a third node n_3. The weight spread from n_1 to n_2 over the indirect path via n_3 is influenced by the (number of and weight of) edges of n_3. This is undesirable since the weight of the direct edge from n_1 to n_2 already completely describes the relationship between n_1 and n_2. Moreover, the influence of the triangle-spreading is likely to cascade the effect of the deeper graph on final tag weights, since the indirect path along which the undesired spread happens has a length of only two hops.

4.2.2 PathRank

In order to address the swash-back and triangle spreading problems we present our adapted weight-spreading approach for undirected folksonomy graphs, which we call PathRank. Rather than doing iterative weight spreading, PathRank assigns a score to each node in the graph based on the shortest path(s) from the preference nodes. Node weights are first computed independently for each preference node and then a weighted average of the resulting weights is taken for each node in the graph to produce the final node weights, taking into account all preference nodes. The weight spreading computation works in a similar manner to a breadth-first search, where edges which were already explored in previous iterations are not re-visited. PathRank is akin to spreading activation which is usually applied to directed graphs, and where nodes spread their weight only once. However, PathRank is used on the un-directed folksonomy graph and gives the edges a personalised direction starting from the query nodes, where the edge direction can be different for each query. PathRank can thus be described as activation-directed weight spreading. In contrast to the original iterative weight spreading approach of FolkRank, PathRank only uses personalised weights, originating from each of the preference nodes, and there are no general importance weights in the graph. Because of the separate calculation per preference node and the absence of general importance weights, each individual weight-spreading calculation has only one node, the preference node, from which all of the weight in the graph originates. The swash-back and triangle spreading of weights can then be prevented by adapting the iterative weight spreading algorithm with a simple rule: *If the weight of a node has been updated in a previous iteration (i.e. is not equal to zero), then do not re-calculate the node's weight.* This methodology is illustrated in Figure 4.7 which depicts the PathRank weight spreading algorithm for one preference node. User u_1 is the query user and so the node representing u_1 in the graph is set as the preference node from which all of the weight originates. At the start, before the weight spreading begins, all of the edges in the graph are undirected. In the first iteration of weight spreading, u_1 spreads weight to all nodes which are directly connected to it. These make up the 1-hop neighbourhood of u_1, labelled as h1. In the next iteration, the nodes in the 1-hop neighbourhood spread weight, however, they only spread to nodes which have not been encountered yet. For example, document d_1 spreads weight to u_2, t_4 and t_5. However, d_1 does not spread back to u_1, preventing swash-back, and also does not spread to t_1, t_2 or t_3, which prevents triangle spreading of weights. All of the nodes reached in the second iteration make up the 2-hop neighbourhood (h2) for u_1, meaning that the shortest path(s) from u_1 to any of the nodes in h2 consist of 2 hops.

The weigh-spreading is then continued in this manner, always exploring one hop further into the graph from the preference node.

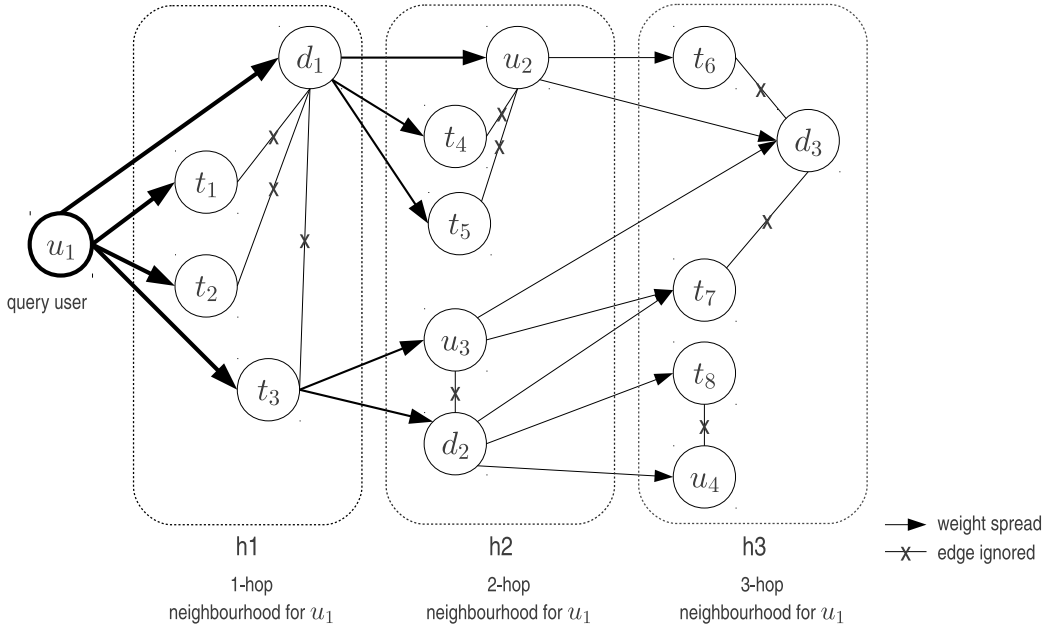


Figure 4.7: PathRank Weight Spreading

The PathRank weight spreading algorithm is in effect not an iterative update calculation like in PageRank/FolkRank, but rather assigns a weight to each node n_i based on the edges of the shortest path(s) from the each of the preference nodes n_p to n_i. The parameter pl specifies the maximum path length from the preference nodes to be explored by PathRank. The end condition of PathRank weight spreading is that either the maximum path length pl has been reached, or that all nodes in the graph have been explored and assigned a weight greater than zero. If we set pl to five for example then all tag nodes which are five or less hops away from the preference nodes will receive some weight and will be considered as candidates, whereas tags which are further than five hops away will not be included.

The benefits of PathRank are that the problems of swash-back and triangle-spreading of weights are removed, which allows the algorithm to fully utilise the information contained in the deeper graph. Since there are no general importance weights, these also cannot interfere with and cascade the influence of the weight spread through the deeper graph. Intuitively we would assume that weights spread from preference nodes through the deeper graph would result in a better re-ranking of the tag nodes in comparison to using general importance weights, since the general

importance of nodes is not personalised and constant across all query posts. Setting different values for the maximum path length pl to be explored allows for a direct evaluation of the value of including the deeper graph in the recommendation process. In our evaluation in Section 4.3.2 we address the question of how much value there is in exploring each step deeper into the graph when calculating tag predictions.

Regarding runtime, as long as we only have one preference node, the complexity of weight spreading is greatly reduced in PathRank compared to FolkRank, since once a node's score is set it does not need to be re-calculated in every subsequent iteration. If we take the same graph, let i denote the total number of iterations and n denote the number of edges in the graph, FolkRank's iterative weight spreading has a complexity of $\mathcal{O}(2n \cdot i)$. In each iteration, weight is spread in both directions along each edge, partly because the nodes are initialised with random starting weights. PathRank has a worst-case complexity of $\mathcal{O}(n)$ if the weight spreading is performed until all nodes in the graph are explored. Weight is only spread once along each edge in one direction. However, in the case that there are several preference nodes, PathRank needs a separate weight-spreading calculation for each of them, meaning the complexity would be $\mathcal{O}(n \cdot p)$ where p is the number of nodes with weight > 0 in the preference vector, whereas the runtime of FolkRanks's iterative algorithm would not change. For the expensive FolkRank algorithm to be applicable in practice, the individual tag scores per user and per document have to be pre-calculated offline, and then combined in the online recommendation phase to quickly generate predictions. In this scenario, where each of the pre-calculation runs has only one node in the preference vector, PathRank is guaranteed to outperform FolkRank regarding runtime. Moreover, by limiting the maximum path length via the parameter pl, the runtime can be further reduced. As we show in the evaluation, pl can be set to almost minimal values without a decrease in prediction accuracy.

4.3 Evaluation and Results

We evaluate our adapted graph models and weight spreading methods with and without the inclusion of content data. The content source in all of the content-aware approaches is the document title and the content inclusion method is to include similar documents in the preference vector. Analogous to SimFolkRank we apply this method of including content to the PathRank algorithm to give the content-aware SimPathRank. In our evaluation we compare the proposed graph models and weight spreading methods in order to answer the research questions given below. We first run experiments on the evaluation set with default parameter settings for

the dampening factor in FolkRank approaches ($d = 0.5$), path length in PathRank approaches ($pl = 10$), and balance between query user and query document in all approaches ($b = 0.5$). We then evaluate the impact of each of these parameters in detail, and finally give results on the real test set with optimal parameter settings. The datasets used are the same as in the last chapter, described in Section 3.4.

Graph Models

- Which of the examined graph models provides the most accurate representation of the tagging data?

Weight Spreading Methods

- Is iterative weight spreading worth the computational expense?
- Do general importance weights provide a benefit?
- Does exploring the deeper folksonomy graph provide an improvement to tag predictions?

4.3.1 Graph Models

Post Graph Scores Retrieval Method

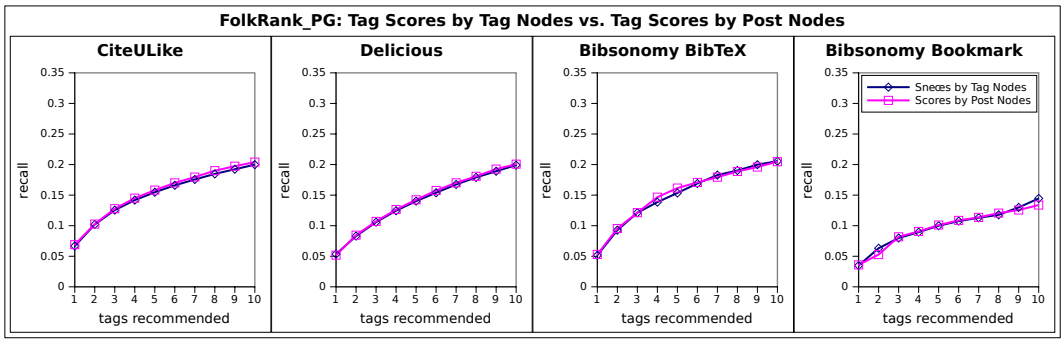


Figure 4.8: Post Graph Scores Retrieval Method

Before comparing the graph construction methods we first evaluate the two alternative scores retrieval methods of the Post Graph model described in Section 4.1.3. The approach of retrieving post node weights from the graph and then calculating the tag scores based on these gives slightly better results than retrieving the tag node weights directly from the graph, although it does not seem to make a significant difference. Since it also makes sense that the number of tags in each

post should not influence the scores of the tags they contain, we use the strategy of calculating tag scores from post nodes for all approaches using the Post Graph model in the subsequent experiments.

Post Graph vs. Folksonomy Graph Models

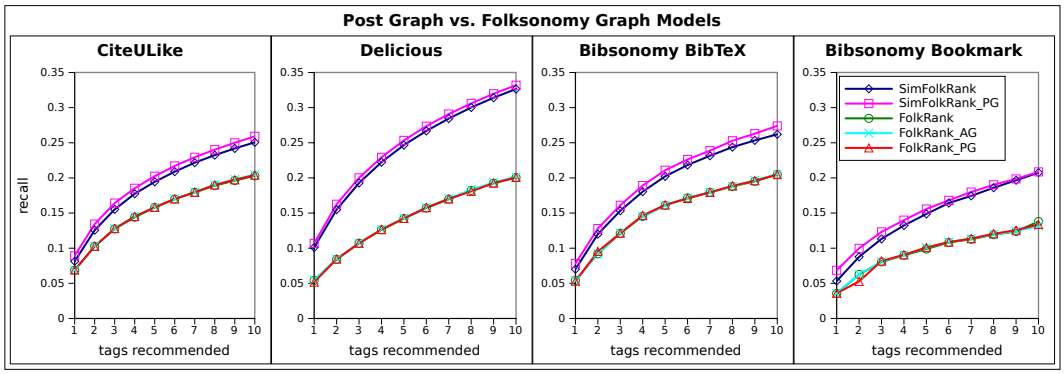


Figure 4.9: Post Graph vs. Folksonomy Graph Models

As shown in Figure 4.9, without content data there is no real difference in results with the different models, and the folksonomy graph (FolkRank), Adapted Graph (FolkRank_AG) and Post Graph (FolkRank_PG) give almost identical results. However, when including content data, the Post Graph model performs consistently better than the folksonomy graph, indicated by SimFolkRank_PG performing better than SimFolkRank across all datasets. We believe the improved results to be due to the more accurate data representation of the Post Graph model, as discussed in Section 4.1. With more nodes in the preference vector, the implicit assumptions of the folksonomy model have a relatively greater impact on tag predictions scores and the Post Graph proves to be the more robust model.

4.3.2 Weight Spreading Methods

Iterative vs. PathRank Weight Spreading

We compare the iterative spreading algorithm of FolkRank to our PathRank weight spreading approach on the folksonomy graph (Figure 4.10) and the Post Graph model (Figure 4.11). The two weight spreading methods produce very similar results on both models across all datasets. However, PathRank is a much quicker weight spreading algorithm. It does not adjust the weight of each node in several iterations to find the optimal distribution of weights reflecting the overall edge

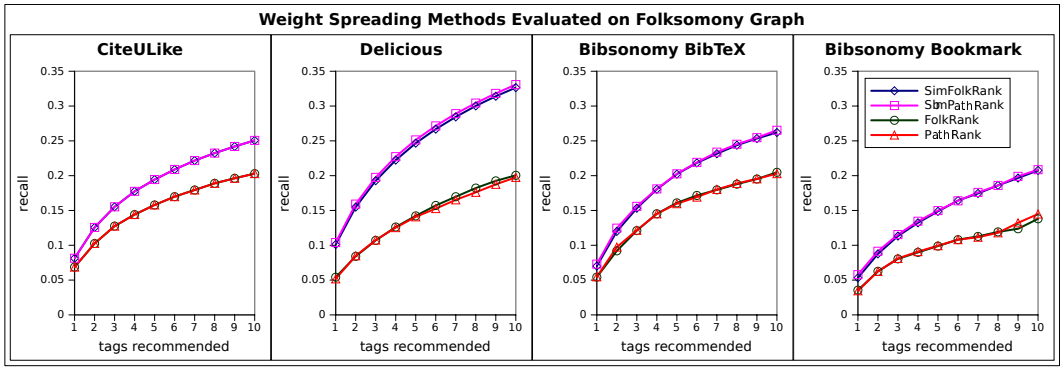


Figure 4.10: Iterative vs. PathRank Weight Spreading on Folksonomy Graph

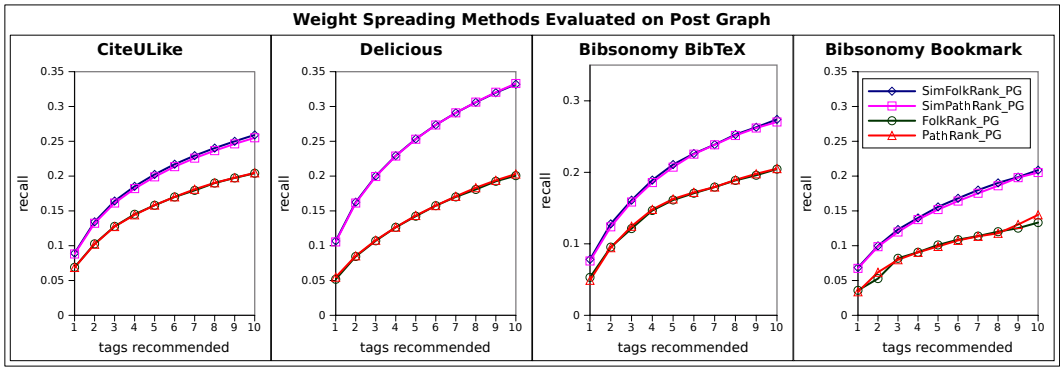


Figure 4.11: Iterative vs. PathRank Weight Spreading on Post Graph

connections in the graph. In other words, it does not consider the general (non-personal) importance weight of nodes which is implied by the graph structure itself. This suggests that the impact of the general importance (or authority) of nodes in the graph does not provide a significant benefit to the tag predictions, and the expensive iterative spreading of the non-personalised weights can be omitted to speed up the recommendation process. Our evaluation of the dampening factor in the next subsection further confirms this conclusion as the best results with FolkRank's iterative weight spreading are achieved at the lowest setting for d, which translates to giving the least relevance to general importance weights.

Value of General Importance Weights

To examine the value of including the general node weights in the recommendation process, we evaluate different settings for the dampening factor d and give our results in Figure 4.12. Without the inclusion of content data there is not much impact on

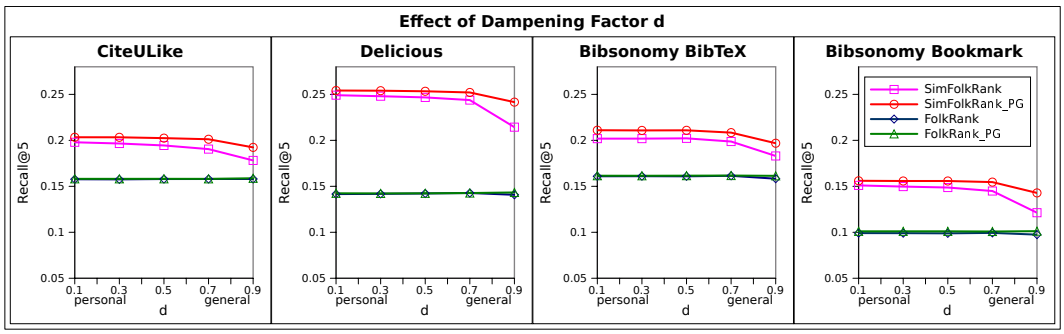


Figure 4.12: Effect of Dampening Factor d on Recall@5

the results for the examined values of d. This is because without content the whole preference weight is given to a maximum of two preference nodes, the query user and document, which means that there is a huge difference in weight between the preference nodes and any one of the other nodes in the graph. Non-preference nodes, and thus general importance weights, don't have a chance to impact the predictions except for extreme values of d such as 0.9, at which setting we observe a very slight decrease in results. With content data the preference weight is distributed among a maximum of 101 preference nodes, which include the query user and potentially 100 training documents similar to the query document. Here the impact of the general non-personalised weights can be observed at lower values of d. In all cases, the best results are achieved with setting d to the lowest examined value of 0.1. This indicates that the general weights in the graph do not provide a benefit to the accuracy of tag predictions, and in fact have a negative impact when given too much relevance. We conclude that to maximise the tag prediction accuracy, d should be set to the lowest value, in effect ignoring the general/non-personalised weights of nodes in the graph. With the lowest examined setting of $d = 0.1$, the general weights can still act as tie-breakers for tags in the candidate set which have otherwise equal personalised weights. However, our results in the comparison with PathRank weight spreading, which does not utilise general weights, suggest that there is no significant improvement over randomly ranking tags which have equal weights. This comparison is made in the previous subsection and in the evaluation on the real test set with tuned parameters in Section 4.3.4.

Predictive Value of Deep Graph

The parameter of maximum path length pl in our PathRank weight spreading approach is especially interesting since it allows us to examine the value of exploring

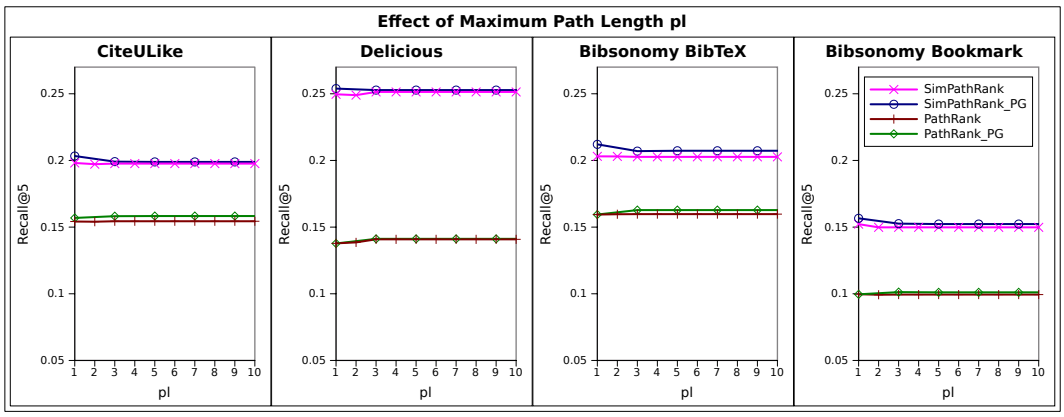


Figure 4.13: PathRank: Effect of Different Settings of Maximum Path Length *pl*

the graph beyond the immediate neighbourhood of the query user and nodes related to the query document. We show the outcome of setting different values of *pl* in Figure 4.13 on the folksonomy graph and Post Graph models. The x-axis gives the value of *pl* and the y-axis is recall@5. With the lowest setting of *pl* only the immediate neighbourhood is explored, where as we move to the right of the x-axis longer paths are also traversed by the weight spreading algorithm. With the Post Graph model we retrieve tag scores as the sum of weights of post nodes they are connected to. Here, the next posts and thus additional tags beyond the immediate neighbourhood (of path length 1) are encountered at a path length of 3. Overall, our results suggest that there is actually not much value in considering the graph beyond the immediate neighbourhood of the preference nodes. There is a small difference that can be observed between path lengths 1 and 3 which we explore in detail below. Moreover, with the PathRank weight spreading algorithm, we have now removed the other influences on the weight spreading calculation which could have cascaded or reduced the impact of the deeper graph. Even without swash-back, triangle-spreading of weights and general importance scores, the weights spread through long paths in the deep graph do not provide a significant improvement. The results indicate that the deeper graph does not provide a beneficial re-ranking of existing candidate tags in the immediate user or document neighbourhood. With the setting of $pl = 1$ where only the immediate neighbourhood is considered, tag nodes which have equal weight will be ranked randomly in the final predictions. Utilising the deep graph to re-rank these tags does not significantly improve results over this random ranking.

Our first detailed observation is that after the first influence of the deeper graph at path length 3, we cannot observe any significant impact, positive or neg-

ative, caused by exploring longer paths. Along the lines of [Wetzker et al., 2010], this suggest that users of (broad) folksonomies have a highly personal tagging behaviour. It is thus very difficult to traverse more than a few edges in the graph and still weigh the encountered nodes in a manner relevant to the the preference node at which the path started. The only small change that can be observed is up to a path length of three. As a side note, the Post Graph model gives better results than the folksonomy graph at a path length of one. At this setting the only difference in the tag scores calculation between the two models is that for the Post Graph the tag scores are given as the sum of weights of post nodes they are connected to. As discussed in section 4.1, this follows from the Post Graph model's assumption that the number of tags of each post should not influence tag scores, whereas the plain folksonomy graph assumes that if there are many tags in a post then each of them is less important. This again suggests that the assumptions made by the Post Graph model provide a more accurate representation of the underlying social bookmarking data.

Another interesting observation in Figure 4.13 can be made from the results with the folksonomy graph model on the Delicious dataset. In this case there is a small improvement at a path length of 3. What is interesting here is that the increase does not occur at $pl = 2$ but at $pl = 3$. In the folksonomy graph, the tags found at a path length of 2 have paths of the form $u_p \rightarrow d \rightarrow t$ or $d_p \rightarrow u \rightarrow t$ from the user preference node u_p or the document's preference node(s) d_p respectively. Including these additional tags is conceptually similar to tag expansion via the document or user nodes related to the preference node. At a path length of 3, paths of the form $u_p \rightarrow t \rightarrow \{u \vee d\} \rightarrow t$ and $d_p \rightarrow t \rightarrow \{u \vee d\} \rightarrow t$ are also included which is conceptually similar to performing tag expansion by using tag-tag co-occurence measure. The small improvement in prediction accuracy seems to be due to using tag-tag co-occurence, rather than giving weight to tags which are related to non-tag nodes from the preference node's immediate neighbourhood. On the BibSonomy Bookmark dataset we can observe a small decrease at $pl = 2$ when including content with SimPathRank. With the Post Graph model and content (SimPathRank_PG), there is also a decrease on BibSonomy as well as CiteULike when going from $pl = 1$ to $pl = 3$. As there are no paths with length 2 leading to additional tags in the Post Graph, the influence of tag expansion both via non-tag nodes and tag-tag co-occurrence is included at the same time at $pl = 3$. It seems to be the case that tag expansion via non-tag nodes decreases results. Along the lines of our discussion in Section 4.2, this seems to suggest that tags found related to non-tag nodes of the preference node but not directly connected to the preference node itself should not

be given an increased weight. As they seem to worsen results it might be appropriate to decrease their weight instead. This suggest a potential that negative feedback could be extracted via a more complex analysis of the graph, which we intend to investigate in the future.

Overall, we conclude that spreading weight into the deeper graph does not provide a significant benefit to tag recommendations and can in some cases even harm prediction scores. The only increase in scores is given by spreading weight from tags to further tag nodes, essentially performing a tag set expansion via tag-tag co-occurrence. Given the complete graph model this is very difficult to separate from expanding the tag set via non-tag nodes, which seems to decrease prediction accuracy. To still utilise the tag-tag co-occurrence data we believe that separate approaches which directly model the tag-tag relationships would be more appropriate and produce better results. However, even though the assumptions made by conventional positive-reinforcement weight spreading methods do not seem to hold for the social bookmarking domain, some useful information could potentially be gained from the deep folksonomy graph by different approaches. A rule-driven analysis of small subsections of the graph could be used to make deductions about implied negative feedback, to either aid the recommendation process directly or to improve the accuracy of a tag-tag similarity metric by including negative scores.

4.3.3 Balance Between Query User and Query Document

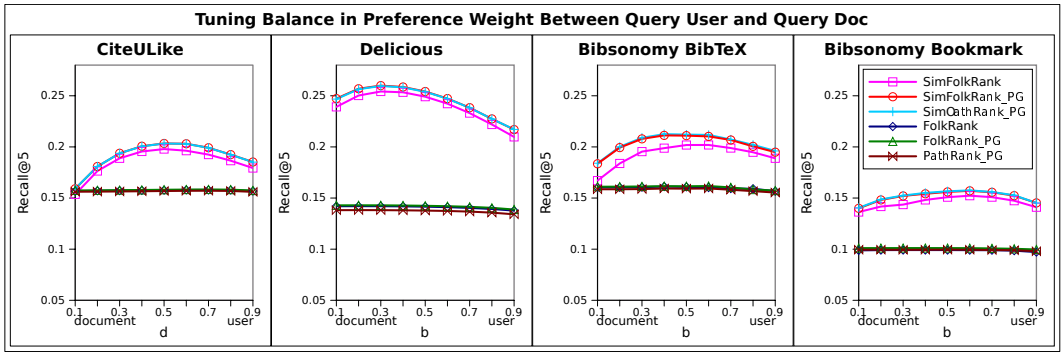


Figure 4.14: Balance b Between Query User and Query Document

In Figure 4.14 we present the results for different settings of b, which determines the balance in preference weight between the query user and the query document. Once again there is not much difference in results without including content data (FolkRank, FolkRank_PG, PathRank_PG). Since most of the query

documents in the test sets are new, the preference vector without content will only include the query user in the majority of cases. For the cases where the document does exist in the graph, and thus will be included in the preference vector, each of the tags connected to the query document will usually receive more weight than each of the tags connected to the query user since users are usually connected to many more tags than documents are. The tags connected to the query user only have a chance to outweigh the tags connected to the query document for high values of b, at which settings we see a slight decrease in results. However, with content data (SimFolkRank, SimFolkRank_PG, SimPathRank_PG) the preference vector contains the query user as well as several documents related to the query document and we can clearly observe the impact of b. The best results are achieved with setting b to 0.5 for CiteULike, 0.3 for Delicious, 0.4 for Bibsonomy BibTeX, and 0.6 for BibSonomy Bookmark.

4.3.4 Results on Test Set

Here we present our final results with optimal parameter settings on the test set of each of the datasets. The content source in all of the content-aware approaches is the document title. For approaches using FolkRank's iterative weight spreading the dampening factor is set to $d = 0.1$, and for approaches using PathRank the maximum path length is set to $pl = 1$. The balance b in preference weight is set per dataset to the best value that was found in the parameter tuning runs.

Figures 4.15 and 4.16 show the recall and F1 respectively, on the test set for each of the datasets. The results on the test set are in line with our previous conclusions on the evaluation set. SimFolkRank_PG produces better results than SimFolkRank over all datasets, suggesting that the Post Graph is a more accurate model of the tagging data than the folksonomy graph. Furthermore, PathRank_PG and SimPathRank_PG give almost equivalent results to FolkRank_PG and SimFolkRank_PG respectively which suggests that the iterative computation and general importance weights in FolkRank's weight spreading approach do not provide a significant benefit to tag predictions. While producing comparable results, the PathRank weight spreading method is much less computationally expensive. Furthermore, the results with PathRank_PG and SimPathRank_PG are achieved with a parameter setting of $pl = 1$. At this setting only the immediate neighbourhood of preference nodes is considered. None of the approaches improve results by utilising the deep graph over SimPathRank_PG with $pl = 1$ which has the information scope of a co-occurrence recommender (SimCoOcc) at this setting. The results on BibSonomy Bookmark without including content data are due to the fact that a large

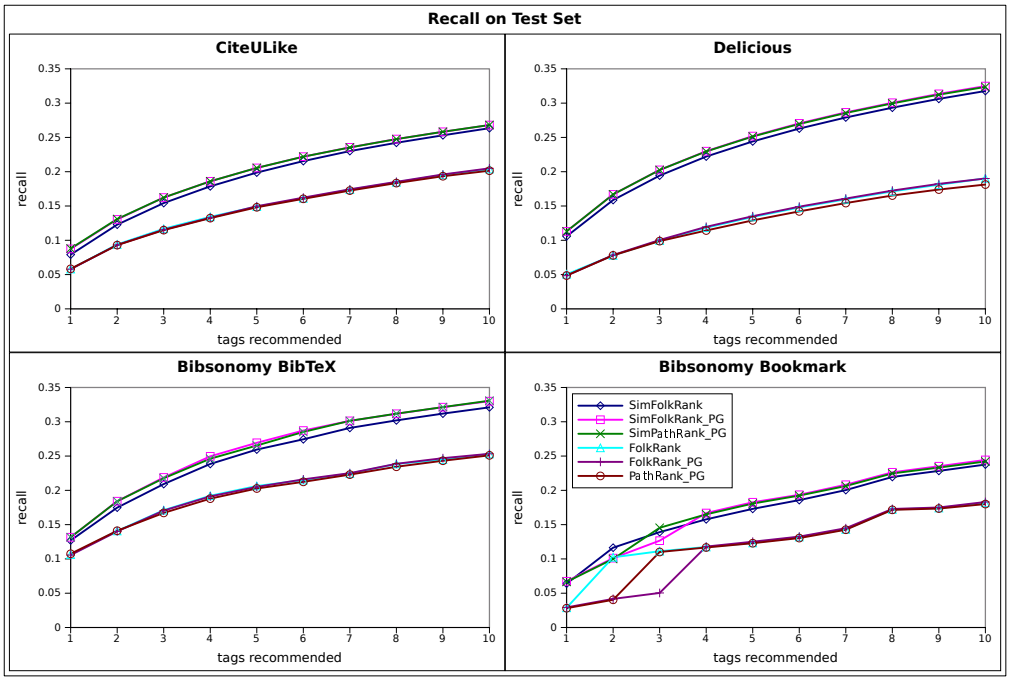


Figure 4.15: Recall on Test Set with Tuned Parameters

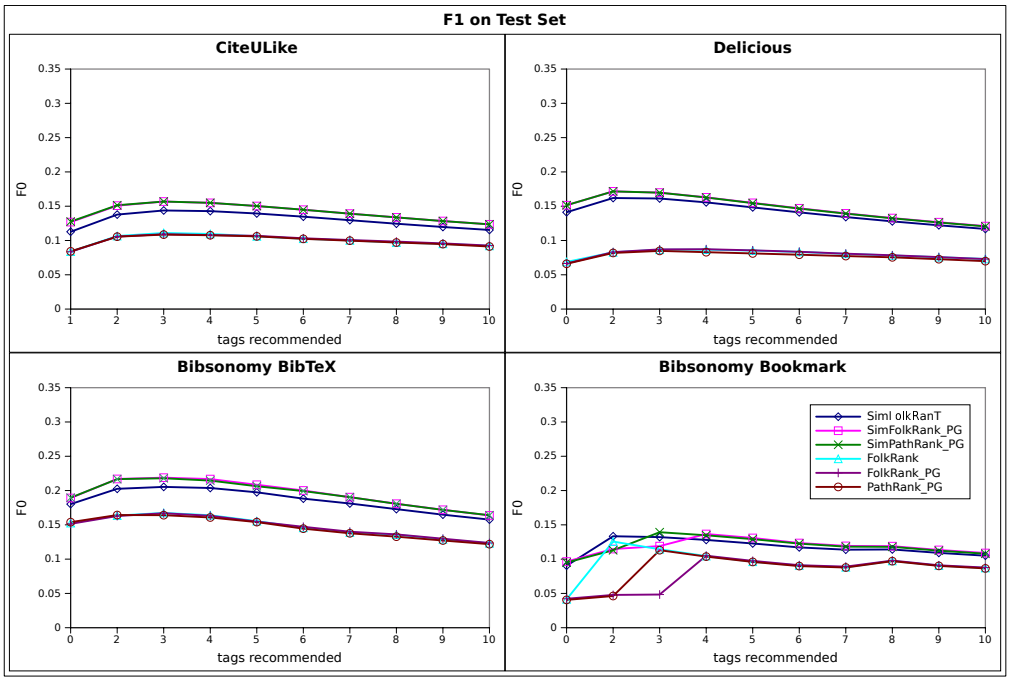


Figure 4.16: F1 on Test Set with Tuned Parameters

portion of the test posts in BibSonomy Bookmark contain new users as well as new documents. For these test posts the algorithms which do not include content data (FolkRank, FolkRank_PG and PathRank_PG) default to recommending the overall highly-ranked tags in the graph without personalisation. The three approaches have different rankings for the top three tags in the general recommendations which leads to the results shown.

Results on Post-Core 2

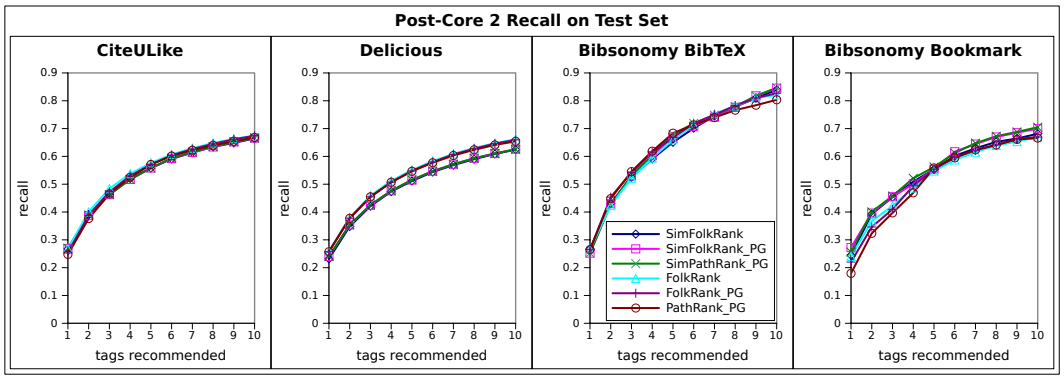


Figure 4.17: Post-Core 2 Recall on Test Set with Tuned Parameters

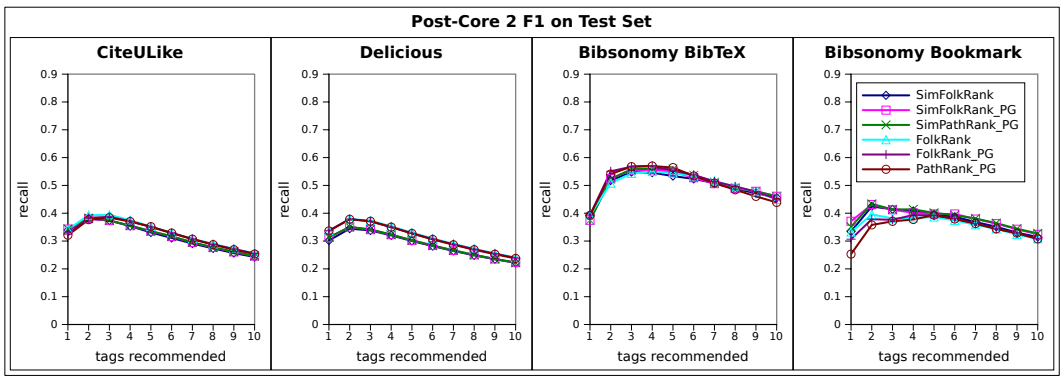


Figure 4.18: Post-Core 2 F1 on Test Set with Tuned Parameters

Figures 4.17 and 4.18 show the recall and F1 of our approaches on each of the datasets' post-core at level 2. The parameters are tuned on the post-core 2 evaluation set and the results shown here are on the test set with optimal parameter settings. As on the unpruned datasets, we have found the best parameter settings to be $d = 0.1$ for the dampening factor in FolkRank weight spreading approaches,

and $pl = 1$ for the maximum path length in PathRank. As discussed in the previous chapter, there is no clear benefit in including content for post-core 2 datasets. The two graph models, Post Graph and regular folksonomy graph, give similar results. The two weight spreading approaches FolkRank and PathRank also produce very similar results. As on the unpruned datasets, there seems to be no benefit in including the deeper folksonomy graph beyond the immediate neighbourhood of query nodes into the recommendation process.

4.3.5 Comparison with Co-Occurrence Recommenders

For completeness, we show the direct comparison in F1 of the best graph-based approaches with the best co-occurrence approach on the unpruned datasets in Figure 4.19. These results are on the test set with tuned parameters for all approaches. SimFolkRank_PG and SimPathRank_PG perform equally well to the SimCoOcc recommender, while SimFolkRank using the folksonomy graph model gives worse results. This confirms that the factor that allows the graph-based approaches to now give equally accurate predictions to the co-occurrence approach is the improved Post Graph model. No additional significant benefit is gained by the graph-based SimFolkRank_PG recommender by analysing the full folksonomy graph.

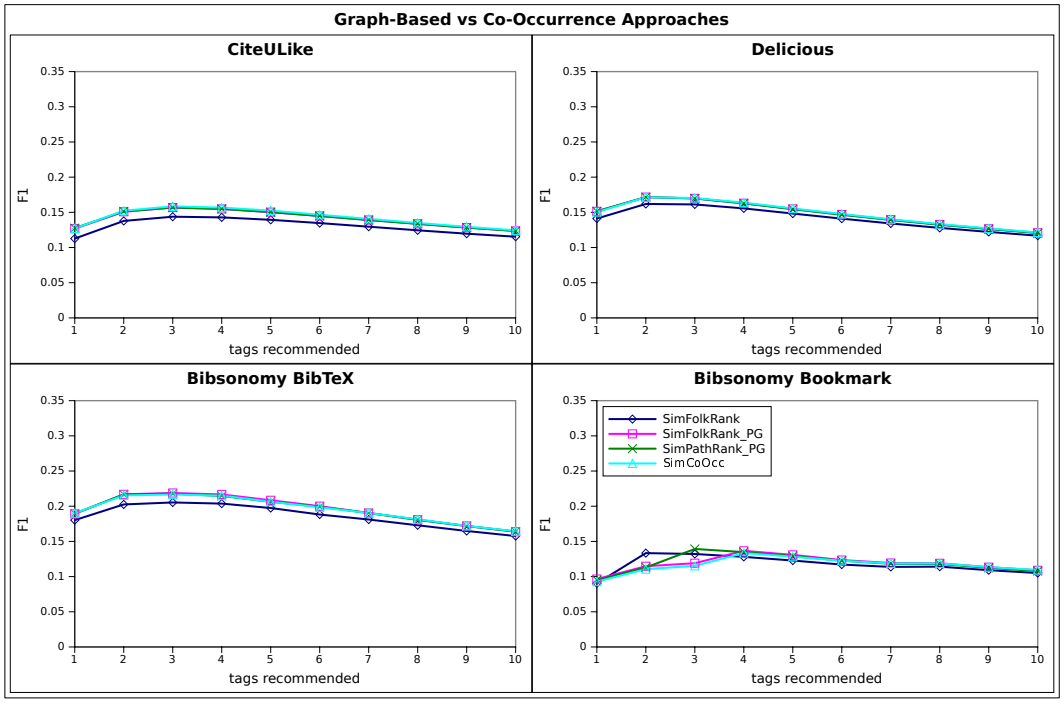


Figure 4.19: Graph-Based vs Co-Occurrence Approaches on Test Set

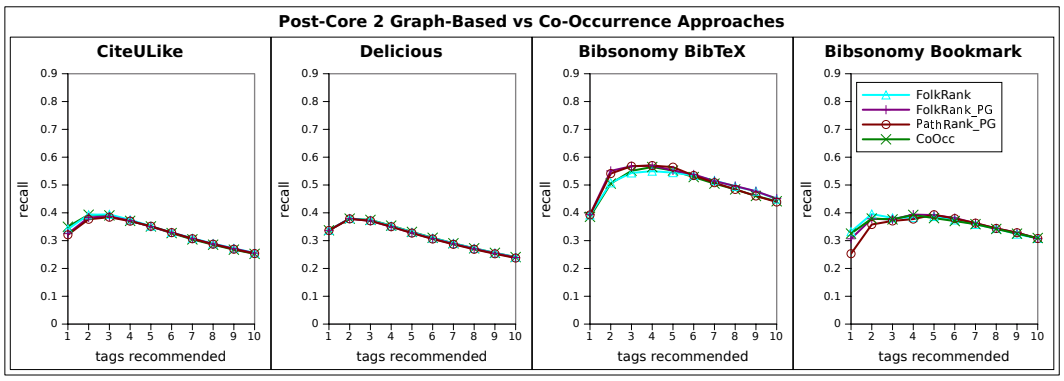


Figure 4.20: Post-Core 2: Graph-Based vs Co-Occurrence Approaches on Test Set

Figure 4.20 shows the comparison of graph-based and co-occurrence approaches on post-core 2. We compare the graph-based methods to the simple CoOcc recommender, where none of the approaches include content data. On post-core 2, all of the methods produce similar results. The only clear conclusion we can make is that using the complex graph-based approaches does not improve recommendation accuracy over the simpler co-occurrence method, as was the case on the unpruned datasets.

4.4 Conclusions

In this chapter we have presented novel adaptations and extensions to FolkRank and conducted an in-depth analysis of the accuracy of the folksonomy graph model, the iterative weight spreading algorithm of FolkRank and the value of exploring the deep folksonomy graph. As part of our examination of the folksonomy graph structure, we have proposed an improved model which captures the tagging data more accurately and produces better tag recommendation results. In our analysis of the iterative weight spreading method of FolkRank, we have shown that the general un-personalised node weights do not provide a positive impact on tag recommendations, and if given too much relevance hurt the accuracy of the algorithm. Since the general node weights are one of the main reasons for FolkRank's high complexity, we think it is an important finding that they can be safely omitted. Furthermore, we have shown that an adapted weight spreading algorithm, PathRank, that works in a similar manner to breadth-first search, produces comparable results to the iterative weight spreading algorithm employed by FolkRank while being computationally less expensive. The most intriguing result of our analysis was that even though both FolkRank's iterative weight spreading and our simpler PathRank spreading algo-

rithm have the potential to utilise the deep folksonomy graph, they do not benefit from doing so in practice. Moreover, we have presented an in-depth discussion as well as a direct evaluation of the value of exploring the deep folksonomy graph. We conclude that exploring the graph beyond the immediate neighbourhood of the query nodes with conventional weight spreading methods does not provide a significant increase in tag recommendation accuracy and can in some cases even hurt recommendations. The assumption that closeness in the graph always implies a positive relationship does not seem to hold beyond the immediate neighbourhood of nodes in social tagging graphs. This suggests that the foundation of graph-based recommenders, which are traditionally applied to two-dimensional datasets, might not apply to the three-dimensional user-document-tag relationships found in social tagging data. In summary our main conclusions are as follows.

Graph Model

- Explicitly including post-membership information into the graph provides a model which makes more accurate assumptions about the relationships in the tagging data and produces improved results over the traditional folksonomy model.

Deep Graph Exploration

- General importance/authority scores, which make iterative weight spreading computationally expensive, do not provide an improvement to the accuracy of tag recommendations and can be omitted to reduce complexity.
- The expensive exploration of the deep tagging data graph with conventional weight spreading methods does not provide an improvement to tag recommendations and can in some cases decrease results.
- The assumption that closeness in the graph always implies a positive relationship might not hold in social tagging datasets beyond the immediate neighbourhood of nodes.

In the next chapter we further explore methods to leverage the potential benefit of including the information contained in the deeper folksonomy graph for tag recommendation. We think that by using rule-based methods which analyse smaller subgraphs of the folksonomy, implicit negative feedback could be extracted. This could be used to include negative scores in user-tag and especially document-tag relationships in order to reduce the scores of tags which are likely to be incorrect for a specific user or document.

Chapter 5

Negative Feedback

In this chapter we further analyse the characteristics of social tagging data and examine how the deeper folksonomy graph can be utilised to aid the generation of tag recommendations. We propose that social tagging data has some unique characteristics and that the interpretation of relationships between nodes in the folksonomy graph has to be adjusted to account for this fact. Conventional weight spreading approaches such as FolkRank assume that only positive feedback exists in the graph. Generally speaking, nodes in the graph are assigned a positive weight which is inversely proportional to their distance (number of hops) from the query nodes. Nodes close to the query nodes get a high positive weight while nodes further away get a smaller positive weight. The base assumption is that the closer two nodes are in the folksonomy graph, the stronger their positive relationship. We suggest that this assumption does not hold for the social tagging domain, and that some of the multi-hop paths in the graph actually indicate a negative relationship. We propose novel negative feedback measures for social tagging data and evaluate these in order to show the existence of implicit negative feedback in the folksonomy graph. Our work reveals important insights into the fundamental characteristics of social tagging data.

While negative feedback is a well-addressed issue in recommender systems that deal with two-dimensional relationships, it has not yet been been explored in detail for social tagging data, which is three-dimensional. By exploiting the three-dimensional relationships, our metrics are based on more accurate assumptions than the traditional recommender systems approach of treating all non-co-occurrences in two-dimensional data as negative feedback. The existence of negative feedback in the data has to be considered when designing tag recommendation methodologies in order to be able to leverage the full scope of information contained in the folksonomy.

Using our CoOcc and SimCoOcc recommenders, we evaluate how the negative relationships impact tag recommendations if they are misinterpreted as being positive and show that our metrics lead to a more accurate interpretation of the relationships in the data. The proposed approach can be applied to a variety of recommendation algorithms. In summary we

- show the existence of negative feedback in social tagging data
- propose metrics for measuring negative feedback
- evaluate the impact of misinterpreting negative feedback as positive

5.1 Existing Approaches

In traditional item recommendation systems that deal with 2-dimensional user-item relationships, explicit negative feedback can be present in the data through negative or low ratings. Alternatively, implicit negative feedback can be assumed to be indicated by an absence of co-occurrence, where a user has not added an item to his collection. However, the implicit negative feedback approach is less accurate since it makes the assumption that the user is aware of the item and has consciously decided not to add it. In tag recommendation an analogous approach has been applied by [Heymann et al., 2008], where all non-co-occurrences between documents and tags are interpreted as negative training examples for a classifier. Similarly, [Rendle et al., 2009] argue that implicit negative feedback is included in the hyper-graph model of [Symeonidis et al., 2008], and that all non-existing tag assignments $(u, d, t) \notin A$ are treated as indicating a negative relationship between the corresponding user u, document d and tag t. [Rendle et al., 2009] improve upon this interpretation by highlighting that additionally to positive and implicit negative feedback, there is also cases which should be treated as having an absence of evidence and thus not be included as negative examples. To differentiate between missing feedback and implicit negative feedback, an evidence-based approach with regard to posts is used. A negative relationship is only interpreted from a non-existing tag assignment $(u, d, t) \notin A$ if user u has a post in which he tagged document d with some tags other than t, that is at least one tag assignment $(u, d, t') \in A$ exists in the data. Thus the approach of [Rendle et al., 2009] only interprets a non-existent tag assignment as negative given some evidence that there was a possibility for the tag assignment to occur. However, the amount of evidence for negative feedback is still small since the user might not be aware of the tag they did not assign. The user's collection of tagged documents alone does not provide enough evidence to imply

whether the user consciously decided against using the tag or if the tag was just not considered. We propose an approach where more evidence is taken into account to make a more informed decision about implicit negative feedback. By exploiting the three-dimensional relationships of the social tagging data our interpretation extracts more reliable negative feedback from the data. We construct negative relationships between users and tags, and document and tags and quantify the strength of the implicit negative feedback by measuring the probability with which non-existent tag assignment could have occurred.

5.2 Negative Feedback Methodology

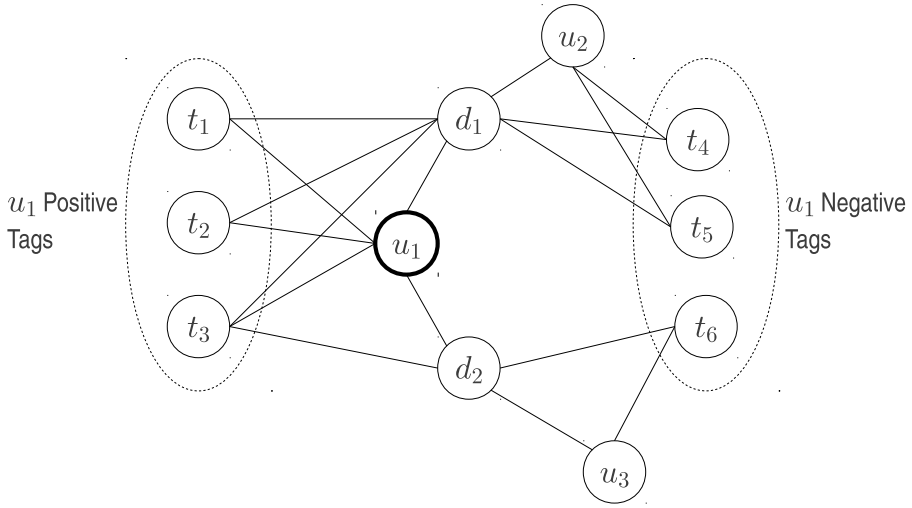


Figure 5.1: Negative Feedback for User u_1

We illustrate our user-related negative feedback methodology in Figure 5.1. User u_1 has tagged document d_1 and with tags t_1, t_2 and t_3, and document d_2 with tag t_3. As the user has used t_1, t_2 and t_3 in the past, these are the positive tags for u_1 and make up the positive User Tags (UT) recommendation set for the user. Document d_1 has also been tagged by another user u_2 with different tags t_4 and t_5. Traditional weight spreading methods would use this information to construct positive relationships between u_1 and t_4, and between u_1 and t_5, as these tags are relatively close to u_1 in the graph. However, we argue that t_4 and t_5 should be given negative weights with regard to user u_1 instead. User u_1 has tagged and, in his view, adequately described document d_1 with tags t_1, t_2 and t_3. He did not require tags t_4 and t_5 to describe the document and used the tags he prefers instead. Since

a different user u_2 did assign those tags to d_1, there is some evidence that these are possible tags for d_1 and thus there was a probability that u_1 could have used the same tags as well. One could argue that u_1 had an opportunity to use tags t_4 and t_5 but consciously chose not to do so which indicates that u_1 has a negative relationship with these tags. Along the same lines, we would also assign a negative weight to t_6 with regard to user u_1. However, the weak point of this argument is that despite the added evidence over previous approaches, u_1 might still not be aware of the tags he did not use, and might want to use them in the future if they are recommended.

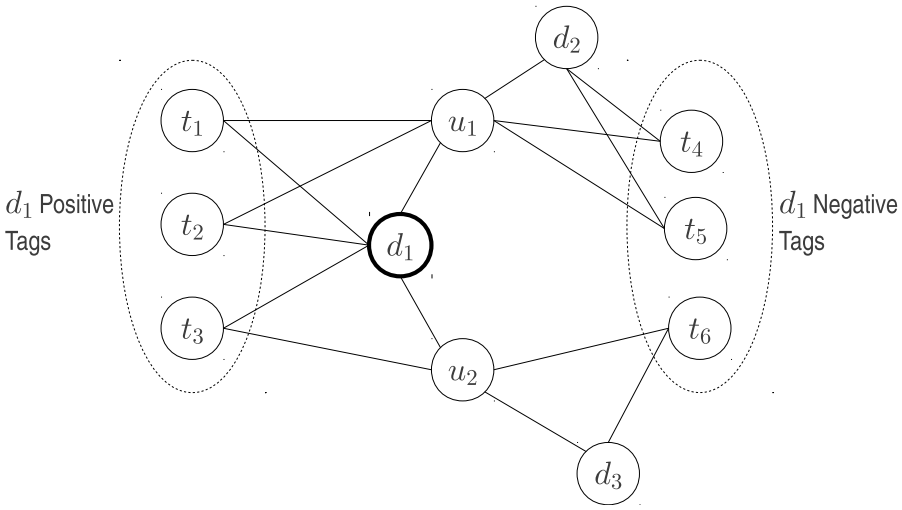


Figure 5.2: Negative Feedback for Document d_1

A much more convincing case can be made for negative feedback with regard to documents. In Figure 5.2, document d_1 has been tagged with t_1 and t_2 by user u_1, and with t_3 by user u_2. These tags are the positive tags for document d_1 and make up the Document Tags (DT) recommendation set. User u_1 has also tagged a different document d_2 with the tags t_4 and t_5, which we classify as having a negative relationship with d_1. Here the argument for negative feedback is much stronger. Since user u_1 has used all of the tags t_1, t_2, t_3 and t_4 in the past, it can be assumed that u_1 is aware of all of the tags' existence. He is using the different tag sets of $\{t_1, t_2\}$ and $\{t_3, t_4\}$ to distinguish between the documents d_1 and d_2. Similarly, user u_2 is using tag sets $\{t_3\}$ and $\{t_6\}$ to distinguish between documents d_1 and d_3, so we also add t_6 to the negative tag set of d_1.

While this interpretation is more plausible than previous approaches, there are some additional factors that are not considered and would be interesting to

explore in the future. The proposed approach assumes that the user is aware of all tags he has used in the past, and that he has described the each document completely, assigning all of the appropriate tags from his tag vocabulary. However, users might be forgetful of their past tagging activity, or might be focused on only certain aspects of the document at the time of tagging. So the tags identified as being negatively related might still be relevant to the document. These issues are lessened to some extent by our our method of aggregating negative feedback and our metrics (described in the next section), however, the time aspect of tagging could also be included in the interpretation of the data. An improved weighting scheme for the negative tags could then be constructed taking recency into account. If a tag is found to have a negative relationship with a document and has not been used in a long time by the user (to tag other documents), then less weight could be given to the negative score of the tag. On the other hand if a tag is identified as a negative and has been used recently by the user for different documents, then the negative score of the tag could receive more weight. The time aspect of tagging and including it in the interpretation of the data is a promising topic that would be worthwhile to explore in the future.

With both our user-related and document-related methodologies there could be cases where positive as well as implicit negative evidence is present for a user or document. For example in the document-related approach if a user has tagged a document d with tag t, and another user has also tagged the same document d but not used tag t despite having t in their tag vocabulary. In these cases we include t only in the positive tag set of d and not in the negative one. The reasoning behind this is that since the interpretation of positive and negative tags for d is aggregated over all users who have tagged d, it would not make sense to allow the implicit negative feedback to further influence the explicit positive feedback. The positive score of each tag included in the positive set already includes the information that only a fraction of the users who have tagged d have assigned tag t to it. The lower the fraction of users that have assigned tag t to d out of all users that have added d to their collections, the weaker the positive score of t will be with regard to d. The sets of tags in our positive and negative groups for a user or document are thus complements without any overlap.

The set of tags included in our negative group corresponds directly to the set of tags found in the 2-hop neighbourhood of the query nodes when using our PathRank ranking algorithm (on the regular folksonomy graph) from the previous chapter. For tag nodes which are further than two hops away from the user or document we make no interpretation as to whether they are negatively related.

This part of our approach is similar to the missing values method of [Rendle et al., 2009], however, less of the tags are interpreted as negative and more tags are left as having missing evidence in our approach.

5.3 Negative Feedback Metrics

Here we define the metrics to quantify the strength of the negative relationships between users and tags, and documents and tags. Even though the relationships are negative we define the weight of the negative relationship as a positive number. In our evaluation we examine the impact of misinterpreting the negative relationships as positives compared to correctly identifying the negative feedback. Please note that while the motivation behind these following metrics is based on characteristics of the folksonomy graph, the metrics themselves are not graph-based approaches. They do not require a graph model of the entire folksonomy to be analysed for every query user or document and do not utilise weight spreading.

5.3.1 Negative User Tags (NUT)

For a query user u_q, the set of tags with negative feedback consists of all tags which were assigned by other users to documents that u_q has tagged but not used by u_q himself (for any document). The weight of the negative relationship for each tag t is given by

$$\mathrm{NUT}(u_q, t) = \begin{cases} \dfrac{\sum_{d_j \in D(u_q)} \mathrm{DT}(d_j, t)}{|D(u_q)|} & \text{iff } t \notin T(u_q) \end{cases}$$

where $D(u_q)$ is the set of documents tagged by user u_q, and $T(u_q)$ is the set of tags used by user u_q (for any document). By including $\mathrm{DT}(d_j, t)$ in the calculation, the strength of the positive relationship between d_j and t is considered as evidence in the negative weight of t with regard to u_q. If t was assigned to d_j in a large fraction of posts related to d_j, but was not assigned to d_j by u_q, then the negative relationship between u_q and t will have more weight. The negative weight is thus quantified by the amount of collaborative evidence that the query user u_q has decided to contradict. If many users have decided to assign tag t for d_j but the query user has decided against using t for this document, we interpret this as u_q having a strong preference against using tag t.

5.3.2 Negative Document Tags (NDT)

The set of tags with negative feedback for a query document d_q consists of all tags that were used by users who have tagged d_q, but were not assigned to d_q itself. We

calculate the negative weight between d_q and each tag t by

$$\mathrm{NDT}(d_q, t) = \begin{cases} \frac{\sum_{u_i \in U(d_q)} \mathrm{UT}(u_i, t)}{|U(d_q)|} & \text{iff } t \notin T(d_q) \end{cases}$$

where $U(d_q)$ is the set of users who have tagged d_q, and $T(d_q)$ is the set of tags assigned to d_q (by any user). The $\mathrm{UT}(u_i, t)$ part includes the strength of the positive relationship between each user u_i and tag t in the calculation. If u_i has used t in a large fraction of his posts but has not assigned t to d_q, then the negative relationship between d_q and t will have more weight. The rationale behind this is that if a user has been using tag t for many of his documents then t is a tag that he prefers to use, however, since he has not assigned t to d_q this is a strong indication that t is not an appropriate tag for d_q.

5.3.3 Negative Similar Document Tags (NSDT)

Following our content inclusion methodology, in order to identify and calculate negative tag weights for documents that have previously not been tagged we apply our similar documents approach. Analogous to SDT, the positive Similar Document Tags, we calculate the negative tag scores for a query document d_q from the negative tag scores of similar training documents.

$$\mathrm{NSDT}(d_q, t) = \sum_{d_j \in N(d_q)} \mathrm{sim}(d_q, d_j) * \mathrm{NDT}(d_j, t)$$

where $N(d_q)$ is the neighbourhood of training documents similar in content to d_q, and $\mathrm{sim}(d_q, d_j)$ is the similarity between d_q and d_j. However, we then remove all tags from NSDT that are included in the positive SDT. This ensures that a negative score is set only if tag t is not in the positive tag set of any similar document $d_j \in N(d_q)$, and that the tag sets of SDT and NSDT are complements.

5.4 Combination Methods

Here we give an overview of the combination methods we use to evaluate the impact of negative scores on tag recommendation accuracy. The union was already introduced but we list it here again in a more general form for comparison to the other approaches. The addition and subtraction methods are used in our evaluation as a means to conduct a detailed analysis of our hypotheses regarding negative feedback.

5.4.1 Union (∪)

As mentioned before the union of two recommendation sets includes all tags which appear in one or both of the sets. We use A and B to represent two tag recommendation sets generated by different recommenders, and denote their union as A∪B. The score of each tag t in A∪B is a weighted sum of the tag's scores in A and in B. The effect is that scores of tags which appear in both source sets are increased relative to scores of tags which appear only in A or only in B. The calculate the score of each tag t in A∪B we use

$$\text{A}\cup\text{B}(t) = \begin{cases} b * \text{A}(t) + (1-b) * \text{B}(t) & \text{if } t \in \text{A} \wedge t \in \text{B} \\ b * \text{A}(t) & \text{if } t \in \text{A} \wedge t \notin \text{B} \\ (1-b) * \text{B}(t) & \text{if } t \notin \text{A} \wedge t \in \text{B} \end{cases}$$

where $\text{A}(t)$ is the score of t in recommendation set A, $\text{B}(t)$ is the score of t in recommendation set B, and $0 \leq b \leq 1$ is a parameter that determines the balance in importance given to scores in A and B.

5.4.2 Addition and Subtraction of Scores (+/−)

Additionally to the union, we introduce two new combination method which we denote by A+B and A−B. Instead of including all tags that appear in one or both of two source recommendation sets A and B, the aim of these methods is to adjust the tag scores of existing tags in A with the scores of matching tags in B. The tags in recommendation set A are re-ranked based on their scores in B;s however, no additional tags are added from B that do not exist in A. The addition and subtraction operations can thus be thought of as a left join instead of a union. In the addition A+B, tag scores from B are added to existing tag scores in A. The score of each tag t in the resulting set is given as

$$\text{A}+\text{B}(t) = \begin{cases} b * \text{A}(t) + (1-b) * \text{B}(t) & \text{if } t \in \text{A} \wedge t \in \text{B} \\ b * \text{A}(t) & \text{if } t \in \text{A} \wedge t \notin \text{B} \end{cases}$$

where $\text{A}(t)$ is the score of t in recommendation set A, $\text{B}(t)$ is the score of t in recommendation set B, and $0 \leq b \leq 1$ is the balance in importance given to scores in A and B. If a tag t is not in A, then it is also not included in A+B. Similarly, the subtraction operation subtracts from the existing tag scores of A the scores of matching tags in B. However, if the resulting score of a tag is negative due to the subtraction we remove this set from the recommendation completely. We do this in

order to conduct a more aggressive evaluation of our negative feedback methodology by completely removing the tags which end up with a negative weight instead of just ranking them down. The score of each tag t in the subtraction A−B is calculated by

$$\text{A}-\text{B}(t) = \begin{cases} b * \text{A}(t) - (1-b) * \text{B}(t) & \text{if } t \in \text{A} \wedge t \in \text{B} \\ b * \text{A}(t) & \text{if } t \in \text{A} \wedge t \notin \text{B} \end{cases}$$

5.5 Evaluation and Results

The datasets we use for evaluation are the same as before. We evaluate the impact of negative feedback using the CoOcc and SimCoOcc recommenders and their individual parts as baselines. On the full unpruned datasets we use SimCoOcc since content is required to recommend tags for new documents, and on post-cores at level 2 we use the CoOcc recommender without content. We have chosen to evaluate against the co-occurrence methods, instead of the more complicated graph-based recommenders, in order to eliminate all other factors and conduct a straightforward evaluation of the negative feedback methodology. However, the proposed method could also be applied in conjunction with graph-based and other recommendation approaches, which would be a very interesting research direction for the future. The aim of our experiments is to evaluate the validity of our method for identifying negative feedback and the accuracy of our metrics to measure the weight of the negative relationships. We examine the impact of misinterpreting negative tags as having a positive relation to the query and analyse if there is any benefit in doing so. In comparison, we run experiments where the negative feedback is correctly identified and examine whether an improvement in tag recommendation accuracy can be achieved or whether removing negative tags from the recommendation set reduces recall. The following research questions are addressed.

- Is there any value in considering tags from the negative set as candidates for recommendation?
- Is there evidence for the presence of implicit negative feedback in the graph structure of the folksonomy?
- Does misinterpreting negative scores as positives harm tag recommendation accuracy?
- Does the proposed approach provide a reliable method to identify negative feedback?

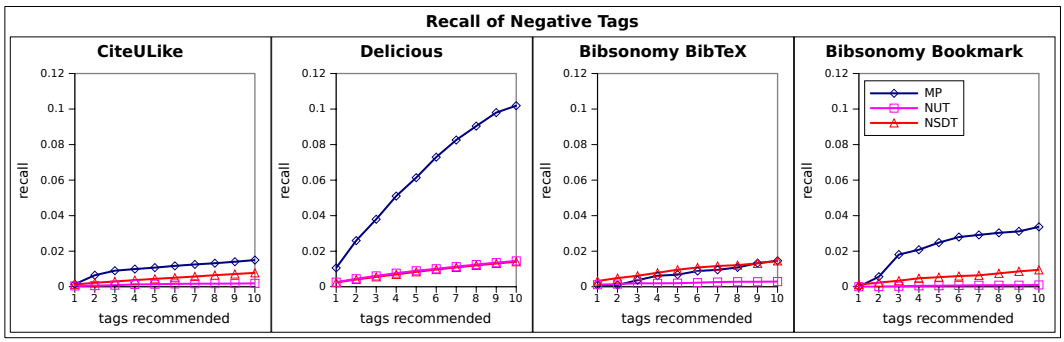


Figure 5.3: Negative Tags as Candidates for Recommendation

5.5.1 Negative Tags as Candidates

The tags we have proposed to have negative relationships with the query user or document are treated as positive tags by conventional weight spreading approaches such as FolkRank and PathRank. These graph-based methods consider the negative tags as candidates for recommendation. In Figure 5.3 we examine whether there is value in considering the negative tags as candidates by comparing the recall of Negative User Tags (NUT) and Negative Similar Document Tags (NSDT) to the recall of recommending Most Popular tags (MP). The negative recommendation sets have very low recall on all datasets. Recommending the same set of most popular tags for all test posts produces better or comparable results than treating the negative tags as a positive recommendation set. The results suggest that in most cases the overall most popular tags should be considered a more valuable source of additional candidate tags than the negative tag set of the query user or document.

5.5.2 Impact of Negative Tags on Recommendations

We then examine the impact on the accuracy of the SimCoOcc recommender when including the tag scores from the negative recommendation sets. Figure 5.4 shows the accuracy of SimCoOcc combined with Negative User Tags (NUT). The union SimCoOcc∪NUT follows the methodology of weight spreading approaches and treats NUT as a positive recommendation set. The tags that appear in both the recommendation set of SimCoOcc and also in NUT are ranked higher relative to tags that only appear in one of SimCoOcc or NUT. Additionally further tags are added to the recommendation set from NUT. The addition SimCoOcc+NUT also treats the Negative User Tags as positive, however, no further tags are added to the recommendation set that do not appear in SimCoOcc. Only the scores of existing tags in

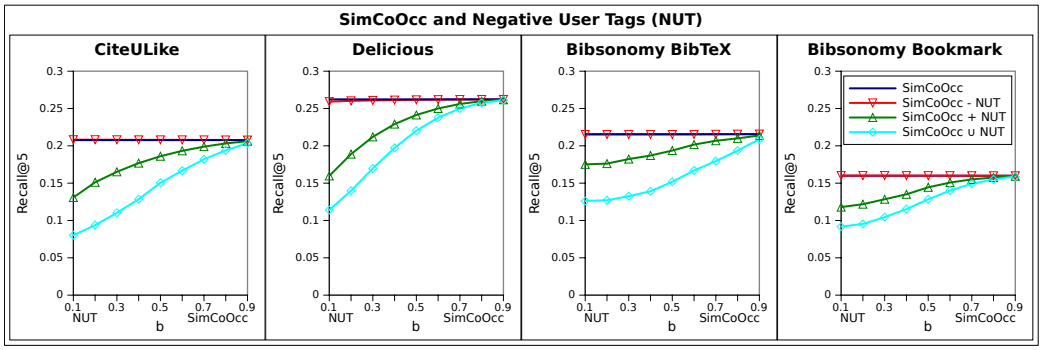


Figure 5.4: Combining SimCoOcc with Negative User Tags

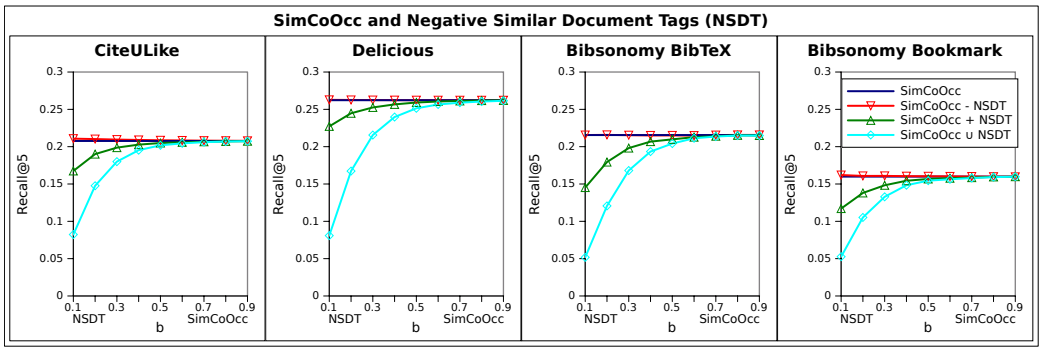


Figure 5.5: Combining SimCoOcc with Negative Similar Document Tags

SimCoOcc that also appear in NUT are increased relative to tags that do not appear in NUT. The third approach is the subtraction SimCoOcc−NUT which treats NUT as a negative recommendation set. The scores in SimCoOcc that also appear in NUT are reduced and if the resulting score of a tag falls below zero the tag is removed completely from the recommendation set. The consequence is that tags with a strong negative rating are not only ranked down to the bottom of the recommendation list but also completely removed. At lower values of the combination weight parameter b, the tag scores in NUT are given the greatest importance, while at higher values NUT plays a lesser role. We can observe that following the methodology of FolkRank and treating NUT as a positive recommendation set, by doing the union SimCoOcc∪NUT, decreases results the more weight is given to NUT, going towards the left of the graph. In SimCoOcc+NUT the Negative User Tags are also treated as positives, but here the assumption is that while including additional candidate tags from NUT is not useful there might still be some value in increasing the scores of tags that appear in the close folksonomy-neighbourhood of the query user. Following this methodology also decreases results the more positive weight is

given to NUT. While the decrease is not as strong as with the union, the addition of the negative scores as positives is disadvantageous at all settings of b. Finally, correctly identifying NUT as negative predictions in SimCoOcc−NUT causes no decrease. The results do not change significantly even at low values of b where a lot of negative weight is given to NUT so that most tags will be removed from SimCoOcc if they appear in NUT. Due to the big decrease observed with SimCoOcc+NUT we can be sure that there is some overlap in the tag sets of SimCoOcc and NUT, and that the tags which appear in both sets will be removed from the recommendation set by the subtraction operation SimCoOcc−NUT with low settings for b. Figure 5.5 shows the same experiment with Negative Similar Document tags, incorporating NSDT into SimCoOcc as positive (SimCoOcc∪NSDT and SimCoOcc+NSDT) or negative (SimCoOcc−NSDT) predictions. The results with NSDT are along the same lines as with NUT. Treating the negative recommendation sets as positive predictions decreases results. This suggests that the assumption made by weight spreading methods that multiple hops in the graph always indicate a degree of positive relatedness does not hold for folksonomies and confirms our conclusions from the last chapter. Correctly identifying and treating NUT and NSDT as negative predictions does not decrease results, however, it also does not give a significant improvement. We can observe that there is some overlap in the tag sets of SimCoOcc and both of the negative tag sets, due to results with the combination method of addition. We suspect that there is no significant improvement in accuracy when subtracting negative scores because the tags appearing in NUT and NSDT are already ranked in low positions in SimCoOcc. Reducing their scores (SimCoOcc−NUT and SimCoOcc−NSDT) thus makes no difference to the top five rankings.

To further explore this issue, we run experiments with the individual parts of the SimCoOcc recommender. SimCoOcc consists of the union of User Tags and Similar Document Tags UT∪SDT (and tags from MP appended to the recommendation set for cases where UT∪SDT cannot produce the wanted number of recommendations). Figure 5.6 shows the impact of including Negative User Tags in the recommendations of Similar Document Tags (SDT) by either subtracting or adding the tag scores in NUT to the tag scores in SDT. In both alternatives, treating the scores in NUT as positives (SDT+NUT) or as negatives (SDT−NUT), only the scores of existing tags in SDT are modified and no additional tags from NUT are added. With the subtraction, tags which end up with a score of less than zero are completely removed from the recommendation set. Similarly, figure 5.7 shows the results of the same experiment with NSDT, subtracting or adding NSDT to UT. As a side note, since the tag sets of User Tags and Negative User Tags are

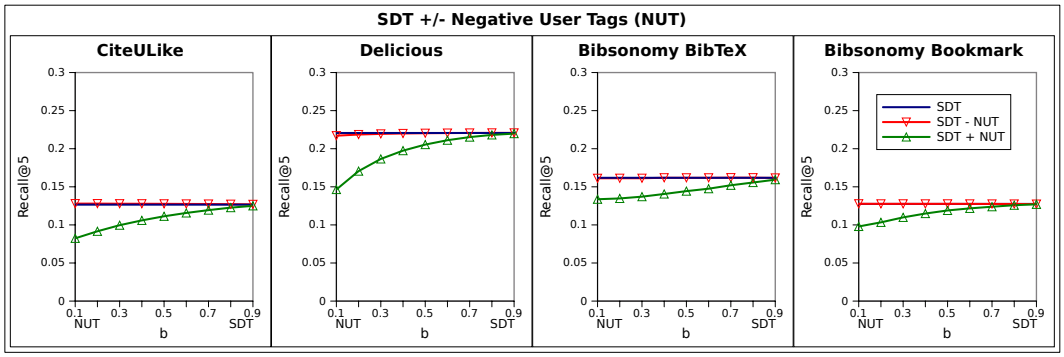


Figure 5.6: Combining Similar Document Tags and Negative User Tags

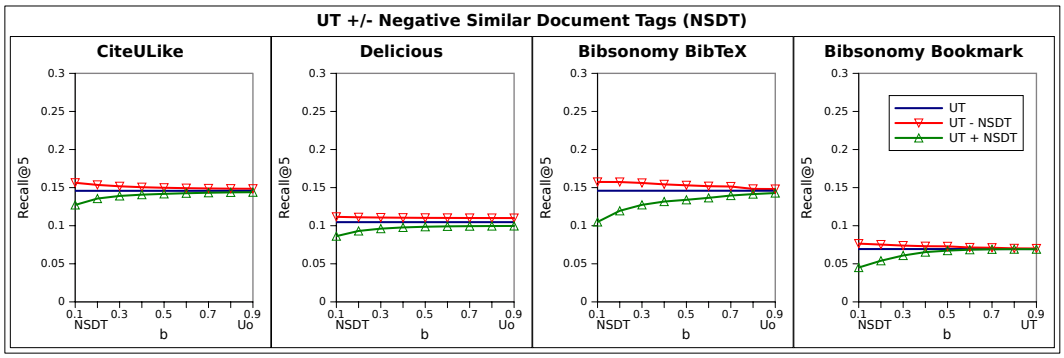


Figure 5.7: Combining User Tags and Negative Similar Document Tags

complements, adding or subtracting NUT from UT would have no effect; and the same holds for SDT and NSDT. As with the previous results, there is a decrease in accuracy when treating the negative tag recommendations as positives with both NUT and NSDT. However, with UT−NSDT there is an improvement over using the prediction scores of UT on their own. As the tags in UT that also appear in NSDT get ranked down they allow other tags in UT which do not appear in NSDT to be ranked higher, resulting in higher recommendation accuracy. We suspect the reason no significant improvement is observed when subtracting NSDT from the complete SimCoOcc recommender is that in SimCoOcc the tags that would be ranked down already have low rankings and are not included in the top recommendations. Since in SimCoOcc the union UT∪SDT is taken, tags that appear in both UT and SDT are likely to be at the top of the recommendations, followed by tags that appear in one of UT or SDT. The effect of the subtraction SimCoOcc−NSDT is that some of the tags in SimCoOcc which do not appear in SDT are ranked down. Since NSDT and SDT are complements only the tags that are not included in SDT are affected

by the subtraction of NSDT. However, due to the union UT∪SDT in SimCoOcc, the tags which do not appear in SDT, and thus only in UT, are likely to be at the bottom of the ranking already and the potential of improvement in accuracy is small. Overall, we conclude that the negative prediction sets NUT and NSDT decrease accuracy if misinterpreted as positives and either improve or do not significantly change recommendation results when treated as negatives. This confirms that the tags in NUT and NSDT indeed have negative relationships with the query user and query document respectively, and implicit negative feedback exists in the folksonomy graph. Our aggressive evaluation of removing tags which end up with a score below zero after subtraction did not harm tag recommendation results, even when a lot of weight was given to the negative scores. This confirms that our methodology of extracting negative feedback has a very low false positive rate and that almost all of the identified negative tags indeed have a negative relationship with the query user or document. Comparing NUT and NSDT, we observe that the extracted negative relationships between documents and tags (NSDT) seem to be more reliable than negative relationships between users and tags (NUT). This is in line with our discussion of the weaker assumptions of NUT in Section 5.2.

5.5.3 Impact on Post-Cores

On post-core 2 datasets we evaluate the negative feedback metrics applied to the CoOcc recommender without content inclusion. The results are in line with our previous conclusions and are shown for completeness in Figures 5.8 and 5.9. Misinterpreting the negative relationships decreases tag recommendation accuracy also for cases where all query document have been previously tagged and content information is not included.

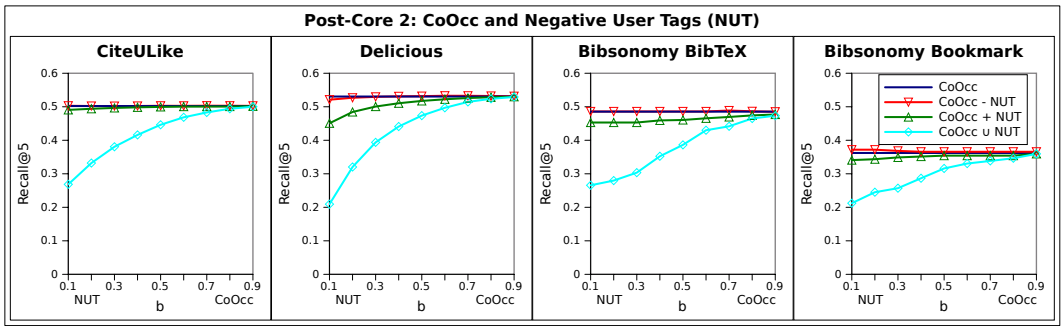


Figure 5.8: Post-Core 2: Combining CoOcc with Negative User Tags

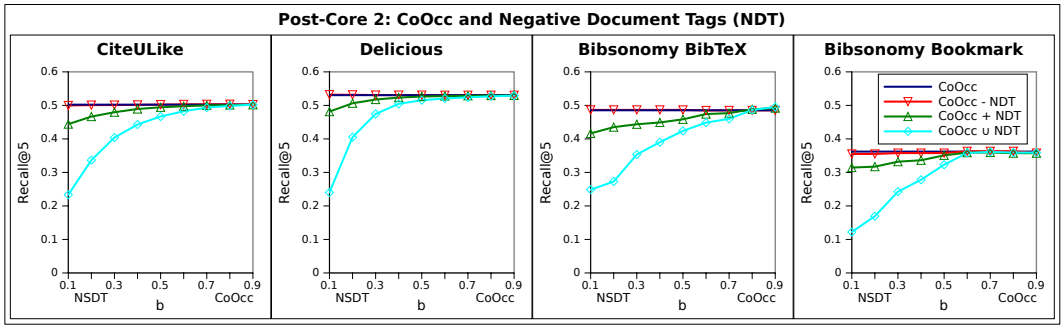


Figure 5.9: Post-Core 2: Combining CoOcc with Negative Document Tags

5.6 Conclusion

We have proposed an evidence-based method to identify and extract implicit negative feedback from the graph and suggested metrics to measure the strength of the negative relationship between nodes. Our evaluation confirmed our suspicion that negative feedback exists in the data structure of the folksonomy and that it can harm tag recommendation results if misinterpreted as positive feedback. This is an important insight that has wide-reaching implications. In the previous chapters we had shown that even though the graph-based FolkRank as well as our adaptations have the potential to utilise the full scope of information contained in the social tagging data they do not benefit from doing so in practice, and no improved results over simpler co-occurrence measures are achieved. After showing the existence of negative feedback in the folksonomy we can conclude that the accuracy of graph ranking algorithms is impaired due to their misinterpretation of the relationships in the social tagging data. While the weight spreading along some of the longer paths in the folksonomy graph could be beneficial and allow graph ranking methods to produce improved results, the negative feedback is encountered at the first level of depth beyond the immediate neighbourhood. Since the paths with misinterpreted negative feedback are shorter than any of the other multi-hop paths in the graph, the decrease in accuracy that is caused by their influence cascades over any potential benefit that could be gained from spreading deeper into the folksonomy. In order to leverage the full scope of the folksonomy in an advantageous manner, the negative feedback in the data has to be taken into account by graph ranking approaches. The existence of negative feedback is a fundamental characteristic of social tagging data and our proposed methodology could be applied to a variety of recommendation algorithms, including graph-based as well as other methods, in order to increase their accuracy.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis gives an overview of existing tag recommendation algorithms and proposes novel approaches to address the new document problem and the task of ranking tags for recommendation. An in-depth analysis of graph ranking methods for tag recommendation is conducted and fundamental characteristics of social tagging datasets are revealed. Existing tag recommendation approaches can be categorised based on the source and scope of information they consider. The information source in folksonomy-based approches is the social tagging data, and in content-based approaches it is mainly the document content. While approaches based solely on the social tagging data achieve a high accuracy when recommending tags for existing documents, content-based approaches are required to address the new document problem and generate tag suggestions for documents that have previously not been tagged. An important characteristic and difference between various folksonomy-based algorithms is the scope of tagging data that is considered when generating recommendations for a query post. Co-occurrence approaches consider only the immediate co-occurrence information of the query user and query document, collaborative filtering and tag expansion approaches have a wider information scope, and graph-based approaches have the potential to analyse the full information scope of the folksonomy with regard to the query.

In order to address the new document problem we have presented two methods to include content data into the recommendation process of folksonomy-based algorithms. In contrast to existing hybridisation methods for tagging data and content, our approach was to include content into folksonomy-based approaches at the algorithmic level. By including content information into graph ranking ap-

proaches we have produced a recommendation methodology that can utilise the full information scope of the folksonomy while also being content-aware. To generate recommendations for a query post our algorithms consider not only the query user and query document, but also the content of the document as part of the query. Our two approaches for including content were to include content words directly into the data model of the folksonomy-based recommenders, or to include content information indirectly by using document similarity. We have applied and evaluated these two alternatives with the graph-based FolkRank and adaptations thereof, as well as simpler co-occurrence recommenders for comparison. Furthermore, two different sources of content data were evaluated with regard to their value in aiding the recommendation process. Our content-aware recommenders achieved a significantly higher recommendation accuracy than their solely folksonomy-based counterparts in experiments using unpruned, real-world test data. Including content at the document level via content similarity was found to be a better content inclusion method than breaking up documents into their individual words and building models at the word level. Even though the word-based approach produced comparable results on larger social tagging datasets it is computationally more expensive, and had less accuracy than the document-based method on smaller datasets. Out of the two examined sources of content data we have found using the document title to produce better results that using the full text content of documents. Additionally, the document title is a much smaller document representation that requires less computational expense. In our experiments with post-core datasets where (almost) all query documents have previously been tagged we have found no clear benefit in additionally including content into the recommendation process. However, there is potential for improvement of the approach to consider such cases as we discuss in the suggestions for future work following this section.

To address the problem of ranking tags we have focused our work on graph ranking algorithms, in particular FolkRank, since these methods have the potential to utilise the full scope of information available in the folksonomy. We have conducted an in-depth analysis of the ranking approach of weight spreading algorithms and highlighted particular issues that arise when graph ranking methods are applied to the tag recommendation task. Our research included detailed examinations of graph models for social tagging data as well as the weight spreading algorithms themselves. We have exposed implicit assumptions made by the widely-used tri-partite graph model of the folksonomy and discussed how they can affect tag rankings and decrease recommendation accuracy. To overcome the issues we have suggested an improved graph model that includes an additional type of node, explicitly repre-

senting posts in the graph. Our Post Graph model was shown to provide a more accurate representation of the social tagging data and improve the tag recommendation accuracy of graph ranking algorithms. Analysing the ranking algorithm itself, we have highlighted issues in FolkRank's computation of tag ranks and proposed an adapted weight spreading approach for social tagging graphs. Our adapted algorithm, PathRank, produced comparable results to FolkRank's tag ranking approach while being computationally much less expensive. Moreover, an important discovery of our work came from directly comparing the accuracy of graph ranking approaches to simpler co-occurrence ranking methods. Our initial intuition, in line with the general consensus of the research community, was that graph ranking methods such as FolkRank would produce more accurate tag recommendation results than simpler approaches that only consider one level of co-occurrence. This intuition was based on the fact that FolkRank has the potential to utilise the full scope of available information while co-occurrence recommenders only consider the direct co-occurrence of the query nodes. However, we have shown in our work that conventional graph ranking approaches do not achieve significantly better results over co-occurrence methods for tag recommendation. Our evaluation of weight spreading with different setting for the scope of information that is considered revealed that the computationally expensive exploration of the deeper graph does not provide a benefit to tag recommendations. Even though graph ranking has the potential to harness the full scope of available information, we concluded that conventional weight spreading methods do not mange to leverage this information in a beneficial way for tag recommendation.

To further explore the tag ranking problem and analyse why graph ranking did not improve results over simple co-occurrence we considered the possibility that negative feedback might exist in the graph structure of the folksonomy. The base assumption of weight spreading is that closeness in the graph always indicates a positive relationship between nodes, with the amount of positive relation decreasing over distance. This assumption holds in other problem domains where the underlying data from which the graph is constructed is two-dimensional, such as PageRank for links between websites. However, whether or not this is true for social tagging data had not been previously examined. To address this issue we have analysed the data structure of folksonomies and suggested that some relationships between nodes in the tagging data graph are in fact negative. We have proposed an approach to identify implicit negative feedback in the graph structure of the folksonomy and introduced metrics to measure the strength of the negative relationships. Our approach exploits the tree-dimensional data structure of folksonomies to extract a more

informed and reliable indication of implicit negative feedback than would be possible for two-dimensional datasets in traditional recommender systems. In our evaluation we have shown that strong indications of implicit negative feedback are present in the folksonomy and that this can harm the accuracy of tag recommendations if the negative relationships are misinterpreted as being positive. Our results confirmed that implicit negative relationships in folksonomies can be accurately identified and have to be taken taken into consideration when designing tag ranking algorithms in the future. We believe this to be an important finding concerning the fundamental characteristics of social tagging datasets that could be used to improve the accuracy of many existing tag recommendation approaches. Overall, the main findings of our work are summarised as follows.

Content-Awareness

- Including content into the tag recommendation process addresses the new document problem and significantly increases results on unpruned real-world datasets.
- The title of documents is a more reliable content source and provides a more accurate description of documents than the fulltext content.
- Including content at the document level produces more accurate recommendations than including content at the word level, especially for smaller sized social tagging datasets.

Graph-Based Ranking

- Explicitly including post-membership information into the graph provides a model which makes more accurate assumptions about the relationships in the tagging data and produces improved results over using the traditional folksonomy graph.
- The expensive exploration of the deeper tagging data graph with conventional weight spreading methods does not provide a significant improvement to tag recommendations.
- The assumption that closeness in the graph always implies a positive relationship does not hold in datasets from the social tagging domain.

Negative Feedback

- Strong indications of implicit negative relationships exist in the graph structure of the social tagging data and have to be taken account when designing recommendation algorithms.

- Negative relationships between users and tags, and documents and tags, can be identified by analysing the paths between the corresponding nodes in the folksonomy graph.
- The three-dimensional social tagging data allows for informed and reliable metric for measuring implicit negative feedback to applied.

6.2 Suggestions for Future Work

The findings presented in this thesis open up many interesting possibilities for future work concerning the inclusion of content and the improvement of graph-based ranking methods for tag recommendation. In our content-aware recommenders we have used a basic content representation approach by modelling documents as bag-of-words vectors. This could be improved upon by using more advanced content modelling methods such as topic models or applying approaches from natural language processing to better identify important words. In our experiments we have found the document title to be the best source of content, however, the full-text content of documents could prove to be valuable in conjunction with a more advanced content model. For test cases where the query documents have been previously tagged and thus have existing tag relationships in the model (post-core level 2), we could not find clear evidence for the benefit of including content data. However, in our content inclusion methodology so far no additional weight is given to the tag relationships of the query document itself if it exists in the model. Weight is given to the tagging profile of the document's content but no special attention is paid to whether or not the exact query document has been tagged before. This approach could be improved upon by introducing an additional weighting between tags found to be related to the exact query document (if it has been tagged before) and tags found to be related to the document's content. If appropriate weights between these can be learned and set, the inclusion of content data could prove to be useful not only for new documents but for already tagged documents as well.

The method for tuning the various parameters is also an interesting research topic. The approach we used was to find one optimal value for each parameter from the evaluation set and then set that value globally for all query posts in the test set. Instead of doing this global tuning across the whole dataset, a user-centred method could also be utilised to find optimal values per user. This would results in a personalisation of not only the recommended tag set but also of the recommendation process itself. For example it could be applied to a parameter which determines the balance in weight between the past tags of the user and tags found to be related to

the title of the document. One user might tend to stick to his own tags that he has used in the past, categorising documents into his own taxonomy of tags. A different user might be more influenced by the titles of documents in his tagging behaviour and choose mostly tags which are related to the title. By tuning the parameter settings per user, a level of personalisation could be achieved that takes into account not only the preferred individual tags of the user but the user's overall tagging behaviour as well. Given sufficient data this approach could then be taken even further by performing an additional personalisation per query, choosing parameters settings not only per user but also considering the particular document that the user is tagging. This could result in for example learning that a user usually prefers tags from his own taxonomy but when the document is outside of his area of expertise he chooses tags from the title.

Concerning graph-based ranking methods we believe there is potential for improvement in order to enable the algorithms to leverage the full information scope of the folksonomy. In our view the most critical improvement would be to adapt the ranking algorithms to consider the negative feedback that is present in the graph structure of social tagging data due to its three dimensions. We have suggested and evaluated a method to identify negative relationships between nodes which could be used to improve existing graph ranking algorithms. One method to achieve this would be to create and include additional edges which carry a negative weight into the graph model. A better approach would be to adapt the ranking algorithms themselves to account for implicit negative relationships. A variety of tag recommendation approaches, graph-based as well as others, could be improved by considering the negative relationships we have identified, and we are excited to see if a significant improvement in tag recommendations can be gained from doing so. For example, negative scores could be included in collaborative filtering approaches where users or documents are represented by a vector of tag weights. In classification approaches our methodology could be used to construct a more reliable set of negative examples. An additional possible application area is the extraction of tag-tag relationships from the folksonomy data. We have proposed and evaluated measures for negative relationships between users and tags as well as documents and tags, however, our approach could be extended for tag-tag relationships. Building on our discoveries about the characteristics of social tagging data, negative relationships between tags could be extracted based on how users use different tags from their tag set to distinguish between documents in their collection. Tag relatedness measures could then be made more accurate by taking into account negative as well as positive feedback.

Another interesting research direction are the sampling methods used in tag recommendation. Sampling methods have been applied to crawl social tagging data where official snapshots are not available. However, the previously applied methods of obtaining sampled folksonomies introduce biases into the datasets. The other application of sampling in tag recommendation is the sampling of training data as a means of data size reduction. As social bookmarking websites and tagging datasets get larger, it is becoming infeasible to build models on all of the training data, especially with computationally expensive approaches which examine complex relationships in the data. It would be interesting to further explore this problem and evaluate different sampling methods in their ability to produce unbiased samples of social tagging data as well as predictive samples of training data for models.

Bibliography

Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender Systems Handbook*, pages 217–253. Springer US, 2011. ISBN 978-0-387-85819-7. doi:10.1007/978-0-387-85820-3_7.

M. Bender, T. Crecelius, M. Kacimi, S. Michel, T. Neumann, J.X. Parreira, R. Schenkel, and G. Weikum. Exploiting social relations for query expansion and result ranking. In *Data Engineering Workshop 2008, ICDEW 2008*, pages 501–506, 2008. doi:10.1109/ICDEW.2008.4498369.

Dominik Benz, Andreas Hotho, Robert Jäschke, Beate Krause, Folke Mitzlaff, Christoph Schmitz, and Gerd Stumme. The social bookmark and publication management system bibsonomy. *The VLDB Journal*, 19(6):849–875, December 2010. ISSN 1066-8888. doi:10.1007/s00778-010-0208-4.

Monica Bianchini, Marco Gori, and Franco Scarselli. Inside PageRank. *ACM Transactions on Internet Technology*, 5(1):92–128, February 2005. ISSN 1533-5399. doi:10.1145/1052934.1052938.

Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, April 1998. ISSN 0169-7552. doi:10.1016/S0169-7552(98)00110-X.

Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, November 2002. ISSN 0924-1868. doi:10.1023/A:1021240730564.

Andrew Byde, Hui Wan, and Steve Cayzer. Personalized tag recommendations via tagging and content-based similarity metrics. In *Proceedings of the International Conference on Weblogs and Social Media*, March 2007.

Fabio Calefato, Domenico Gendarmi, and Filippo Lanubile. Towards social semantic suggestive tagging. In *SWAP: Proceedings of the 4th Italian Semantic Web Workshop*, volume 314 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

Ciro Cattuto, Dominik Benz, Andreas Hotho, and Gerd Stumme. Semantic grounding of tag relatedness in social bookmarking systems. In *Proceedings of the 7th International Conference on The Semantic Web*, ISWC '08, pages 615–631, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-88563-4. doi:10.1007/978-3-540-88564-1_39.

A. Dattolo, F. Ferrara, and C. Tasso. The role of tags for recommendation: A survey. In *Proceedings of the 3rd International Conference on Human System Interactions (HSI)*, pages 548–555, 2010. doi:10.1109/HSI.2010.5514515.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

Folke Eisterlehner, Andreas Hotho, and Robert Jäschke, editors. *Proceedings of ECML PKDD Discovery Challenge 2009, Bled, Slovenia, September, 2009*, volume 497 of *CEUR-WS.org*, 2009.

Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.

Jonathan Gemmell, Andriy Shepitsen, Bamshad Mobasher, and Robin Burke. Personalizing navigation in folksonomies using hierarchical tag clustering. In *Data Warehousing and Knowledge Discovery*, volume 5182 of *Lecture Notes in Computer Science*, pages 196–205. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-85835-5. doi:10.1007/978-3-540-85836-2_19.

Jonathan Gemmell, Thomas Schimoler, Maryam Ramezani, and Bamshad Mobasher. Adapting k-nearest neighbor for tag recommendation in folksonomies. In *Proceedings of the 7th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems*, 2009.

Jonathan Gemmell, Thomas Schimoler, Bamshad Mobasher, and Robin Burke. Hybrid tag recommendation for social annotation systems. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*,

CIKM '10, pages 829–838, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0099-5. doi:10.1145/1871437.1871543.

Minas Gjoka, Maciej Kurant, Carter T. Butts, and Athina Markopoulou. Practical Recommendations on Crawling Online Social Networks. *IEEE Journal on Selected Areas in Communications*, 29(9):1872–1892, October 2011. ISSN 0733-8716. doi:10.1109/JSAC.2011.111011.

Paul Heymann, Daniel Ramage, and Hector Garcia-Molina. Social tag prediction. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 531–538, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-164-4. doi:10.1145/1390334.1390425.

Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in Artificial Intelligence*, pages 289–296, 1999.

Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. Information retrieval in folksonomies: Search and ranking. In *The Semantic Web: Research and Applications*, volume 4011 of *Lecture Notes in Computer Science*, pages 411–426. Springer, 2006a. ISBN 978-3-540-34544-2. doi:10.1007/11762256_31.

Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. Trend detection in folksonomies. In *Proceedings of the First international conference on Semantic and Digital Media Technologies*, SAMT'06, pages 56–70, Berlin, Heidelberg, 2006b. Springer-Verlag. ISBN 3-540-49335-2, 978-3-540-49335-8. doi:10.1007/11930334_5.

Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. FolkRank: A Ranking Algorithm for Folksonomies. In *Proceedings of Workshop Information Retrieval, LWA 2006*, pages 111–114, 2006c.

Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223, Morristown, NJ, USA, 2003. Association for Computational Linguistics. doi:10.3115/1119355.1119383.

D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender Systems: An Introduction*. Cambridge University Press, 2010. ISBN 9780521493369.

Robert Jäschke, Leandro Balby Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in folksonomies. In *Knowledge Dis-*

covery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, pages 506–514, 2007.

Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in social bookmarking systems. *AI Commun.*, 21 (4):231–247, December 2008. ISSN 0921-7126.

Robert Jäschke, Folke Eisterlehner, Andreas Hotho, and Gerd Stumme. Testing and evaluating tag recommenders in a live system. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 369–372, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-435-5. doi:10.1145/1639714.1639790.

Long Jin, Yang Chen, Pan Hui, Cong Ding, Tianyi Wang, Athanasios V. Vasilakos, Beixing Deng, and Xing Li. Albatross sampling: robust and effective hybrid vertex sampling for social graphs. In *Proceedings of the 3rd ACM international workshop on MobiArch*, HotPlanet '11, pages 11–16, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0742-0. doi:10.1145/2000172.2000178.

Sanghun Ju and Kyu-Baek Hwang. A weighting scheme for tag recommendation in social bookmarking systems. In *Proc. the ECML/PKDD 2009 Discovery Challenge Workshop*, pages 109–118, 2009.

Heung-Nam Kim and Abdulmotaleb El Saddik. Personalized pagerank vectors for tag recommendations: Inside folkrank. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, pages 45–52, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0683-6. doi:http://doi.acm.org/10.1145/2043932.2043945.

Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999. ISSN 0004-5411. doi:10.1145/324133.324140.

Christian Körner, Roman Kern, Hans-Peter Grahsl, and Markus Strohmaier. Of categorizers and describers: an evaluation of quantitative measures for tagging motivation. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, HT '10, pages 157–166, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0041-4. doi:10.1145/1810617.1810645.

M. Kurant, A. Markopoulou, and P. Thiran. Towards unbiased BFS sampling. *Selected Areas in Communications, IEEE Journal on*, 29(9):1799 –1809, october 2011. ISSN 0733-8716. doi:10.1109/JSAC.2011.111005.

N. Landia and S.S. Anand. Personalised tag recommendation. In *Proceedings of the RecSys 2009 Recommender Systems and the Social Web Workshop*, pages 83–86, 2009.

Nikolas Landia. Utilising document content for tag recommendation in folksonomies. In *Proceedings of the sixth ACM conference on Recommender systems*, RecSys '12, pages 325–328, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1270-7. doi:10.1145/2365952.2366033.

Nikolas Landia, Sarabjot Singh Anand, Andreas Hotho, Robert Jäschke, Stephan Doerfel, and Folke Mitzlaff. Extending FolkRank with content data. In *Proceedings of the 4th ACM RecSys workshop on Recommender Systems and the Social Web*, RSWeb '12, pages 1–8, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1638-5. doi:10.1145/2365934.2365936.

Chul-Ho Lee, Xin Xu, and Do Young Eun. Beyond random walk and metropolis-hastings samplers: why you should not backtrack for unbiased graph sampling. *SIGMETRICS Performance Evaluation Review*, 40(1):319–330, June 2012. ISSN 0163-5999. doi:10.1145/2318857.2254795.

Marek Lipczak and Evangelos Milios. Learning in efficient tag recommendation. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 167–174, New York, NY, USA, 2010a. ACM. ISBN 978-1-60558-906-0. doi:10.1145/1864708.1864741.

Marek Lipczak and Evangelos Milios. The impact of resource title on tags in collaborative tagging systems. In *HT '10: Proceedings of the 21st ACM Conference on Hypertext and Hypermedia*, pages 179–188, New York, NY, USA, 2010b. ACM. ISBN 978-1-4503-0041-4. doi:10.1145/1810617.1810648.

Marek Lipczak and Evangelos Milios. Efficient tag recommendation for real-life data. *ACM Transactions on Intelligent Systems and Technology*, 3(1):2:1–2:21, October 2011. ISSN 2157-6904. doi:10.1145/2036264.2036266.

Marek Lipczak, Yeming Hu, Yael Kollet, and Evangelos Milios. Tag sources for recommendation in collaborative tagging systems. In *Proceedings of the ECML/PKDD 2009 Discovery Challenge Workshop*, pages 157–172, 2009.

Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages

620–628, Morristown, NJ, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-41-1.

Caimei Lu, Xiaohua Hu, Jung-ran Park, and Jia Huang. Post-based collaborative filtering for personalized tag recommendation. In *Proceedings of the 2011 iConference*, iConference '11, pages 561–568, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0121-3. doi:10.1145/1940761.1940838.

Y. Matsuo and M. Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13:157–170, 2004.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1 edition, March 1997. ISBN 0070428077.

Johannes Mrosek, Stefan Bussmann, Hendrik Albers, Kai Posdziech, Benedikt Hengefeld, Nils Opperman, Stefan Robert, and Gerrit Spira. Content-and graph-based tag recommendation: Two variations. In *Proceedings of the ECML-PKDD Discovery Challenge Workshop*, pages 189–199, 2009.

Martin F. Porter. An algorithm for suffix stripping. *Program: Electronic Library & Information Systems*, 40(3):211–218, 1980. ISSN 0033-0337. doi:10.1108/00330330610681286.

Maryam Ramezani. Improving graph-based approaches for personalized tag recommendation. *Journal of Emerging Technologies in Web Intelligence*, 3(2), 2011.

Maryam Ramezani, Jonathan Gemmell, Thomas Schimoler, and Bamshad Mobasher. Improving link analysis for tag recommendation in folksonomies. In *Proceedings of the 2nd Recommender Systems and the Social Web Workshop at RecSys '10*, pages 39–45, 2010.

Steffen Rendle and Lars Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 81–90, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-889-6. doi:10.1145/1718487.1718498.

Steffen Rendle, Leandro Balby Marinho, Alexandros Nanopoulos, and Lars Schmidt-Thieme. Learning optimal ranking with tensor factorization for tag recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on*

Knowledge Discovery and Data Mining, KDD '09, pages 727–736, New York, NY, USA, 2009. ISBN 978-1-60558-495-9. doi:10.1145/1557019.1557100.

Ingrid Renz, Andrea Ficzay, and Holger Hitzler. Keyword extraction for text characterization. In *8th International Conference on Applications of Natural Language to Information Systems*, NLDB 2003, pages 228–234, 2003.

Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March 2002. ISSN 0360-0300. doi:10.1145/505282.505283.

A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke. Personalized Recommendation in Social Tagging Systems using Hierarchical Clustering. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 259–266, New York, NY, USA, October 2008. ACM. ISBN 978-1-60558-093-7. doi:10.1145/1454008.1454048.

Börkur Sigurbjörnsson and Roelof van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, pages 327–336, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-085-2. doi:10.1145/1367497.1367542.

Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C. Lee Giles. Real-time automatic tag recommendation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 515–522, 2008.

Yang Song, Lu Zhang, and C. Lee Giles. Automatic tag recommendation algorithms for social recommender systems. *ACM Transactions on the Web*, 5(1):4:1–4:31, February 2011. ISSN 1559-1131. doi:10.1145/1921591.1921595.

Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 43–50, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-093-7. doi:10.1145/1454008.1454017.

Robert Wetzker, Carsten Zimmermann, and Christian Bauckhage. Analyzing Social Bookmarking Systems: A del.icio.us Cookbook. In *Proceedings of the ECAI 2008 Mining Social Data Workshop*, pages 26–30. IOS Press, 2008.

Robert Wetzker, Carsten Zimmermann, Christian Bauckhage, and Sahin Albayrak. I tag, you tag: translating tags for advanced user models. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 71–80, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-889-6. doi:10.1145/1718487.1718497.

Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. KEA: Practical automatic keyphrase extraction. In *Proceedings of the 4th ACM Conference on Digital Libraries*, pages 254–255. ACM, 1999.

Zhichen Xu, Yun Fu, Jianchang Mao, and Difu Su. Towards the semantic web: Collaborative tag suggestions. In *Proceedings of the Collaborative Web Tagging Workshop at WWW 2006*, Edinburgh, Scotland, 2006.

Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, ICML-97, pages 412–420, 1997.

Valentina Zanardi and Licia Capra. Social ranking: uncovering relevant content using tag-based recommender systems. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 51–58, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-093-7. doi:10.1145/1454008.1454018.

Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. World wide web site summarization. *Web Intelligence and Agent Systems*, 2(1):39–53, 2004. ISSN 1570-1263.

Yuan Zhang, Ning Zhang, and Jie Tang. A collaborative filtering tag recommendation system based on graph. In *ECML PKDD Discovery Challenge 2009*, pages 297–306, Bled, Slovenia, 2009. CEUR Workshop Proceedings.